PC PLUS

STEP by STEP

IAN
ROBERTSON

MS-DOS 6.0

STEP-BY-STEP

MS-DOS 6.0

Ian Robertson

STEP-BY-STEP

MS-DOS 6.0

Ian Robertson

NEW·TECH

Newtech
An Imprint of Butterworth-Heinemann Ltd
Linacre House, Jordan Hill, Oxford OX2 8DP

A member of the Reed Elsevier group

OXFORD LONDON BOSTON
MUNICH NEW DELHI SINGAPORE SYDNEY
TOKYO TORONTO WELLINGTON

First published 1993
© Ian Robertson 1993

All rights reserved. No part of this publication may be reproduced in any material form (including photocopying or storing in any medium by electronic means and whether or not transiently or incidentally to some other use of this publication) without the written permission of the copyright holder except in accordance with the provisions of the Copyright, Design and Patents Act 1988 or under the terms of a licence issued by the Copyright Licensing Agency Ltd, 90 Tottenham Court Road, London England W1P 9HE. Applications for the copyright holder's written permission to reproduce any part of this publication should be addressed to the publishers.

NOTICE
The author and the publisher have used their best efforts to prepare this book, including the computer examples contained in it. The computer examples have all been tested. The author and the publisher make no warranty, implicit or explicit, about the documentation. The author and the publisher will not be liable under any circumstances for any direct or indirect damages arising from any use, direct or indirect, of the documentation or computer examples contained in this book.

TRADEMARKS/REGISTERED TRADEMARKS
Computer hardware and software brand names mentioned in this book are protected by their respective trademarks and are acknowledged.

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library.

ISBN 0 7506 1726 8

Typeset by P.K.McBride, Southampton
Printed and bound in Great Britain

Contents

Preface 9

- 1 - Introducing MS-DOS

1 What is an Operating System? 12
2 What MS-DOS Does 14
3 Floppy and Hard Disks 15
4 File Actions 19
5 Rename, Directory & Attributes 21
6 Size, Date and Time 24
7 Editing and Printing 26
8 Disk Actions 27
9 Other Actions 30
10 MS-DOS Methods 32
11 Drive Letter and Path 35
12 Summary - Running Programs 38

- 2 - Installation

13 Should You Upgrade? 40
14 Preparing for Installation 42
15 The Installation 45
16 Making System Disks 48
17 The CONFIG.SYS File 50
18 The AUTOEXEC.BAT FIle 53

Contents

- 3 - Common Commands

19 Common Commands 58
20 The DIR Command 59
21 Finding Your Way 64
22 Using TREE 67
23 Creating a Directory 70
24 Removing a Directory 74
25 The COPY Command 76
26 Copying to and from Devices 79
27 COPY Options 82
28 Renaming Files 84
29 Deleting and Moving Files 87

- 4 - DOSSHELL

30 Introduction to DOSSHELL 92
31 The File and Program Option 97
32 File Display Options 99
33 Selecting Files 107
34 File Actions 113
35 Menu Methods 119
36 Program File Running Options 124
37 The Group System 125
38 Creating a Group 134
39 Adding a Program to a Group 137

Contents

40 Running a Program 142
41 Task Swapping with DOSSHELL 143
42 DOSSHELL Options 149
43 Display Options 153

- 5 - Deeper into DOS

44 Commands and DOSKEY 158
45 Using DIRCMD 161
46 Redirection 164
47 Pipes and Filters 167
48 Introducing Batch Files 172
49 A Simple Batch File 178
50 Batch File Parameters 184
51 Using ECHO 187

- 6 - Memory

52 PC Memory 190
53 Memory Reports 192
54 Problems with HIMEM.SYS 196
55 Using EMM386 199
56 Using MEMMAKER 202
57 Hard Disk Utilities 206
58 Making Backups 212
59 Unformatting 217

Contents

60 Undelete Actions 220
61 Disk and File Defragmentation 227
62 Using DBLSPACE 233
63 Using theSMARTDRV Cache 238
64 Options for SMARTDrive 242

- 7 - Viruses and Interlink

65 Viruses 246
66 Dealing with Viruses 249
67 MS-DOS Anti-Virus 252
68 Anti-Virus Options 256
69 Living with Anti-Virus Protection 261
70 InterInk 263

- 8 - Command Summary

71 Other Commands 270
72 APPEND to EXIT 271
73 EXPAND to PATH 275
74 POWER to TIME 279
75 TREE to XCOPY 283
INDEX 285

Preface

MS-DOS 6.0 is the latest in a long line of the essential PC disk operating system from Microsoft. Though MS-DOS has by now been in use for some 12 years, there are still huge numbers of computer users who have still not come to grips with the importance of the system and its commands.

This book is intended as an **introduction** to MS-DOS 6.0 (and much of it is applicable also to MS-DOS 5.0) for the user who feels baffled by the system and intimidated by the large manuals or the multitude of commands and options. A surprising number of MS-DOS users have never found how simple and convenient the DOSSHELL system is, and make no use of it.

Obviously in the space of a book of this size, with full explanation of commands, it would be impossible to cover all the commands and utilities of MS-DOS 6.0. We concentrate in this book on step-by-step illustration of the most important commands and requirements. In addition, the most recent utilities that are packaged with MS-DOS 6.0 are explained.

This means that complications have to be avoided, or methods of dealing with them consigned to references. This is justifiable for some 90% of potential users of MS-DOS 6.0. When you need more information and have the experience to use it you will find that the *Pocket Book of MS-DOS 6.0 (Butterworth-Heinemann)* will provide you with more information. The ultimate reference book is the manual that comes with MS-DOS 6.0 itself, but you need some experience to get what you need from it.

Most users, however, need no more than appears here, and a very large number need only minimal knowledge of the system. It is that minimal knowledge, however, that is most difficult to glean from manuals, because until you have considerable experience you do not know what is important for everyday use and what you might never use.

Preface

Topics such as the installation of a new hard drive, or the manual setting up of a memory manager, have been omitted because they require both experience and expertise. In addition, where a command permits the use of a large number of options, these have been included only if the command is important **and** the options are needed by the average user. The result is, I hope, a book that can be followed, step by step, by the inexperienced user.

Ian Robertson, Spring 1993

- 1 -

Introducing MS-DOS

1 What is an Operating System?

MS-DOS is an operating system, a program that either exists in the computer in the form of a chip (a ROM or Read Only Memory system, used now mainly for portable machines) or which can be loaded in from a disk, using a very small loader program held in ROM. The aim of the operating system is to provide, at the very least, the essential routines that every computer needs to control:

- the screen so that what you type can be seen on the monitor,
- the keyboard so that you can type commands, and
- the disk drive(s), for managing files and loading and running programs.

Without program routines that attend to these tasks, the computer is usable only if each program that is to be run attends to all of these actions for itself. This would be a wasteful duplication, so that the operating system is just as important as all the hardware of the computer.

When micro-computers started to be used seriously for business purposes, many manufacturers devised their own operating systems (OS). The use of different operating systems by different manufacturers, however, makes it virtually impossible to set standards that will allow one computer to run a program that works on another computer from a different manufacturer. In these early days (1970-1980) it was particularly important to be able to control the action of computers that used disk drives, so that a Disk Operating System, or DOS, was essential.

- One solution to the problem of interchangeability came about in the very early days of the development of microcomputers, in the form of **CP/M**.
- This was a system to control, program and monitor (hence the initials) the operation of these early microcomputers, and it was

devised by Gary Kildall in or around 1973, before microcomputers were a commercial proposition and the idea of a standard operating system for micro-computers was quite new.

- Today only a very few portable machines, such as the Amstrad NC100, use their own operating systems.

CP/M came along with a set of utility programs that allowed the user to carry out tasks like file copying, file checking, program modification, disk verification and other housekeeping actions that are now so familiar.

When the original IBM PC was developed, the small firm of **MicroSoft** offered an operating system that allowed more control and more logical commands than CP/M. This started the remarkable growth of MS-DOS and of MicroSoft itself. Since these early days, MS-DOS has become the standard operating system for the PC form of machine. This standardisation has been one of the attractions of the PC, because it has ensured that upgrading the computer did not require upgrading all of the software.

This situation is changing, because modern machines are handicapped by the requirement to be able to run programs that could be run on the original IBM PC. MS-DOS 6.0 is a small step forward to operating systems that will allow the full potential of modern PC computers to be unleashed.

2 What MS-DOS Does

MS-DOS has grown from a small operating system, used mainly by programmers, to a set of commands that are used intensively by millions of computer users all over the world. Though MicroSoft itself has developed more convenient ways of controlling a computer in the shape of Windows, the MS-DOS system remains in charge, because Windows simply makes use of MS-DOS. This means that if you can use MS-DOS directly you can exercise tighter control and make your commands run faster than is possible when there is another program in charge of MS-DOS.

- MS-DOS 6.0 includes DOSSHELL, which is an easier way of making use of MS-DOS commands.

We can group the actions of MS-DOS into sets:

- File actions, which allow you to control the files of programs and data that are held on the hard disk.
- Disk actions that allow you to format and use floppy disks, and which organise the hard disk for easier use
- Utilities which are designed to add convenient methods of carrying out actions that at one time would have required you to be able to program the computer for yourself.
- More advanced methods of controlling the use of memory and the hard disk

It is important to know that MS-DOS controls the PC machine. Many machines now use the MicroSoft Windows system along with programs written specially for Windows. Though convenient, Windows does not replace MS-DOS, and it has to use MS-DOS to carry out its actions. This is rather like asking someone else to use the remote-control for your TV - it might be less effort, but it takes longer, and you are giving up your ability **to control things directly.**

Floppy and Hard Disks 3

In the early days of microcomputers, if a machine had any kind of disk drive at all, it provided only for floppy disks. A floppy disk is so-called because the disk itself is a very thin plastic circle coated on each side with magnetic material, and it is so flexible that it must be encased. The early floppy disks were of 8 inch diameter, but these were replaced by the more familiar 5.25 inch size in the mid-seventies. The casing on these disks is cardboard. Later floppies in the 3.5 inch size used a rigid plastics casing, and these have now almost completely replaced the older 5.25 inch type.

The magnetic recording system that is used on all these disks is very similar to the method used for cassette recorders. The recording process is called **saving** and the replaying process is called **loading**. When you save a file (which can be a program or data) the file is retained on the disk and can be loaded next time you use the computer. This contrasts with anything held in the main memory, which disappears whenever the computer is switched off.

Floppy disks, though almost indispensable, have some severe limitations:

- They can hold only a small amount of data (but see later)
- You need to select which disk to use
- You may need to change from one floppy disk to another in the course of using a program
- You may need several hundred disks after a year or so of using your computer

The amount of data that a disk or the computer memory can hold is measured in units called bytes. A *byte* is the amount used for storing one keyboard character, and there are 256 possible different bytes, more than enough for all the characters on a keyboard. The same bytes are also used for programs, with each byte used as a code for an action.

3 Floppy and Hard Disks

The older 5.25 inch type of floppy can store 368640 bytes, a quantity referred to as 360 Kbyte. The later high-density version can store 1228800 bytes, an amount referred to as 1.2 Mbyte.

- The Kbyte is 1024 bytes. The reason for the use of 1024 rather than 1000 is that the computer counts in twos rather than in tens, and the number 1024 is 2 raised to the power of 10 (1000 is ten raised to the power of 3).
- The Mbyte should be 1024 Kbyte, which is the way that memory units are counted. For disks, however, the Mbyte is often taken as being 1000 Kbyte. This is misleading, but the habit is difficult to change. On this reckoning, the high-density 5.25 inch disks hold 1.2 Mbyte (1200 Kbyte, really 1.17 Mbyte) and the high-density 3.5 inch disks hold 1.44 Mbyte (1440 Kbyte, really 1.406 Mbyte)
- For convenience, these larger disk sizes will be referred to as 1.2 Mbyte and 1.4 Mbyte throughout this book.

In the early days it was usual for a program to fit completely on a floppy disk, sometimes even leaving space for your data files on the same disk. The limitations of floppy disks, even in the larger sizes, soon led to pressure for more disk space. When larger programs were devised they had to be split into sections (called **overlays**), so that you had to change disks to use another part of the program. For example, if you were using a spreadsheet program and needed to draw a graph, you had to insert the graph-drawing disks.

This led to the use of hard disk drives in micro-computers. A hard drive is a collection of aluminium disks on a shaft, spinning ten times faster than a floppy disk and enclosed in a dust-free container. These drives have several important advantages:

- They can have much greater capacities (typically 40 Mbyte to more than 1000 Mbytes).

- They can save and load data very much faster.
- All of your programs and data are available from the moment you switch on the computer without needing to insert any floppy disks.
- You can copy new programs and data to the hard disk from a set of floppy disks. The floppy disks used for programs are called the distribution disks.

Virtually all modern PC machines use hard drives, and MS-DOS 6.0 is intended for such machines. You should not attempt to use MS-DOS 6.0, nor modern programs, on a machine equipped only with floppy drives. The exception is a machine that is networked, meaning that it is connected to a larger machine (the server) which is equipped with a hard drive.

At one time, the copying of program files to the hard drive required a considerable knowledge of MS-DOS methods. Modern programs, including MS-DOS 6.0 itself, are very much easier to install because they come with a SETUP program that carries out these actions for you. At most, you need only answer a few questions and then change from one floppy disk to another when requested.

Recently, floptical drives have become available. These are drives that use a combination of magnetic recording and laser-beam methods (like a CD) to store much more on a 3.5 inch disk. One form of floptical drive can take either a conventional 3.5 inch magnetic disk (1.4 Mbyte) or an optical disk of 21 Mbyte capacity. Others, much more expensive, can use disks of 120 Mbyte or more.

- These drives will become much more important in the lifetime of this book and could possibly replace the conventional type of hard drive with the advantage that the disk can be removed for safe keeping, reducing the danger of damage or fraud.

3 Floppy and Hard Disks

- An alternative to the use of Floptical disks for the smaller sizes is the new high-density drive from Verbatim which can record 24 Mbyte on a 3.5" disk.

File Actions 4

The most important task of any DOS is to allow you to control the file actions of your computer. Files can be of two main types, the **program files** that take control of your computer to allow it to carry out some action such as word-processing; and the **data files** that you create when you run such programs and type in data, words or numbers. Both of these types of files are coded as magnetic signals on floppy disks or on a hard disk, and though the files of a program (sometimes referred to as an **application**) are expensive to buy, the data files are often priceless. This is because they represent your work over a period of, possibly, several years. Your data files might just consist of letters to your Uncle Fred, but they could be the accounts of your business, a set of legal documents, the draft of your first novel; all of them almost irreplaceable. You can buy program files over the counter; your data files are the result of your own hard work.

The three main file actions are:

- Copying, so that you can make an additional copy, called a **backup**, of any valuable file.
- Moving, so that you can shift a file from one part of a hard disk to another, or from one hard drive to another.
- Deleting, so that you can remove unwanted files from the hard disk to allow space for new files.

The **copying** action is the most-used of all MS-DOS actions. You would use it to

- Copy files on the hard disk to floppies as a backup.
- Copy files from floppies to the hard disk.
- Copy files from one part of the hard drive to another.
- Copy files from one floppy to another, possibly a different size or capacity.

4 File Actions

In addition, the command that MS-DOS uses for copying can be used for a variety of other actions, such as printing text or displaying text on the screen. This makes the command one of the most important of all the MS-DOS internal commands.

An **internal command** is one that MS-DOS can carry out without the need to read any additional files from the disk, so that it is carried out faster. Before the almost universal use of hard drives, this was even more important because an internal command could be carried out with any need to replace the disk in the floppy drive.

The **moving** action is, by comparison, much less important. Moving is a combination of copying and deleting, with each file being deleted after it has been copied.

The **delete** action is also very important because you need to be able to:

- Delete old files that are no longer of use to you
- Delete files that have been safely copied and are not immediately needed
- Delete files that are taking up too much space on the hard drive

Rename, Directory & Attributes 5

Renaming a file is an action that is more important than you might think if you have little experience of using files. Its uses are:

- To allow you to use the original name for another file
- To make a file name fall into line with other files in a set
- To avoid wiping out a file by saving another one that is using the same name.

When you save a data file you will be asked to provide a name by which it can be recognised, its **filename**. This filename is subject to rules, one of which is that no more than eight characters are used in the main part of the name (see details later). This can often make it difficult to create a unique name for each file. If you already have a file called MYFILE, then saving another file, and using the same name, will totally replace the original MYFILE with the new file. This is one of the most common causes of the mysterious disappearance of files.

If you rename a file before saving another of the same name (which might be more appropriate for the new file) there is no risk of overwriting the old file. For example, if you rename MYFILE to OLDFILE and then save your new file as MYFILE both files will remain on the disk.

You might also want to make your files more uniform. For example, you might want to rename files called MYFIRST and MYSECOND as CHAP1 and CHAP2 because these formed the first two chapters of a book.

Another way of ensuring that files do not conflict is using directories. A **directory** is a group of files, usually on the hard disk, but you can create and use directories on a floppy disk as well. The principle is to keep sets of files together with the advantages that:

- It is much easier to locate files on a hard drive

5 Rename, Directory & Attributes

- You can have two files with the same name co-existing provided they are in different directories.
- You can obtain a list of files for one directory by itself, as if it were a separate disk.

A hard drive may house several thousand files. This would make it very difficult to find the file you wanted unless they were grouped, and you would also be subject to the usual rules that saving a file would always replace an existing file of the same name. In addition, all disks have a limit to the number of filenames that can normally be used. This is 128 for a hard drive, but a directory full of files counts as just one name, so that if you have 1000 files in a directory (not recommended), this counts as just one name. By using directories you can obtain a list of all the files in one directory, usually a manageable amount, rather than all the files on a disk which could be a very large list.

- There is no limit to the number of files you can place in a directory, other than the capacity of the disk itself.

Another way of handling files which users of PC machines are less familiar with is the use of **attributes**. The attributes of a file are saved and loaded with the file and are all contained in a single byte. They are coded signals, consisting of:

- The *read-only* (**r**) attribute, which stops a file from being erased or altered or another file of the same name recorded over it.
- The *hidden* (**h**) attribute, which prevents a filename from being seen in a listing
- The *system* (**s**) attribute which is reserved for important program files
- The *archive* (**a**) attribute which reveals whether the file has been changed or not since the last time it was saved.

Rename, Directory & Attributes 5

There is also another attribute that is used for the name of a directory, so that the computer can distinguish the name of a directory from the name of a program or data file.

- You seldom need to deal with attributes for yourself, but it is important to know that they exist and why they are used.

6 Size, Date and Time

When you require a list of the files on a floppy, or in one directory of a hard drive, the information that you will normally be presented with is:

- The filename, consisting of main name and extension (see later)
- The size of the file, as a number of bytes
- The date and time of creation or alteration

which are the usual items that you need to know about files. You can also specify that you want less information, such as name only, or more (such as attributes).

Two MS-DOS actions that are not as well-known are concerned with comparing and finding files. You can compare two files and obtain a list of differences between them, if any. This is useful if you have two or more versions of a file and you want to see what has changed from one version to another - it can avoid the need to look through the files trying to find the differences.

You can also use an MS-DOS command to find a text file (not a program file) that contains specified words. You might, for example want to find the file that contains the words *the end of the road* or *80386 microprocessor*. By using the MS-DOS Find command you can locate the file on a floppy or in a hard disk directory.

The file information has another purpose, because it allows lists of files to be displayed in sorted order. The usual order is alphabetical, so that you can see files listed (on the screen or in print) in ascending alphabetical order. Names that start with numerals appear first, followed by names that start with alphabetical letters. You can also opt to have the list in reverse alphabetical order, starting with letter Z and ending with letter A or numerals.

You can also obtain lists that are sorted in other ways. For some purposes, it can be useful to see all of your files listed in order of date

of creation, so that you can see the oldest and the youngest files. You can opt to place oldest first or youngest first.

Another order that can sometimes be useful is size order, so you can see which files are taking up most space on your hard disk.

7 Editing and Printing

MS-DOS contains commands for editing text files and for printing text and other data. Editing is provided by a utility program that is **not** internal, so that when you take this option there will be a slight pause while the hard disk locates the program and starts it running. Editing is an important action because:

- You need it to carry out amendments to a pair of important files called CONFIG.SYS and AUTOEXEC.BAT that control the way that your computer works.
- You need it to create controlling files, called Batch files, which make the actions of locating and using programs much simpler
- You can use it to create short pieces of text that do not need the full resources of a word-processor.

MS-DOS also controls your printer, allowing you to print items like lists of files and the text in text-files. For MS-DOS use it does not matter what type of make of printer you are using as long as it is one that is suitable for use on a PC machine.

When a program, such as a word-processor, spreadsheet or data-base, uses your printer it will usually need to exert more control so as to use effects like bold or italic print, double-spacing, indented margins, different type shapes and sizes (fonts) and even drawing pictures (graphics). MS-DOS does not provide these actions, and separate programs called printer drivers are needed. If your printer behaves itself for purely MS-DOS printing actions (with no other program running) but cannot cope with printing out from other programs then a faulty printer driver may be responsible. Some printer manufacturers supply their own drivers, but not all are good and some are very poor.

Disk Actions 8

The file actions of MS-DOS are the ones that you use most frequently, but there is another set of actions that are specific to disks rather than to files. These include:

- Formatting a floppy disk, and unformatting if required.
- Making a floppy System disk in case of problems with the hard drive
- Copying the complete contents and arrangement of a floppy disk
- Comparing the contents of two floppy disks
- Checking to find if files on a disk or directory are damaged, and to mend them if possible
- Making a coded copy of the contents of a hard disk on a large number of floppies (a full or partial backup of the hard disk)
- Recovering files that were deleted in error from a disk and are still required

All of these are important actions, though they are not necessarily everyday events like copying. A floppy disk is normally bought **unformatted** to allow for being used on different types of machines, though it would be perfectly feasible now that the PC machine is so predominant, to supply disks ready for immediate use.

Formatting means the magnetic marking out of circular **tracks** on the disk, and dividing these tracks into **sectors**, with each sector holding a fixed amount of data, such as 1 Kbyte. A disk, hard or floppy, is not like the old-style gramophone record that has a fixed and visible track. These circular tracks and the division into sectors are magnetic and invisible and can be altered. You cannot use a new disk until formatting has been carried out because there are no tracks or sectors on a new disk.

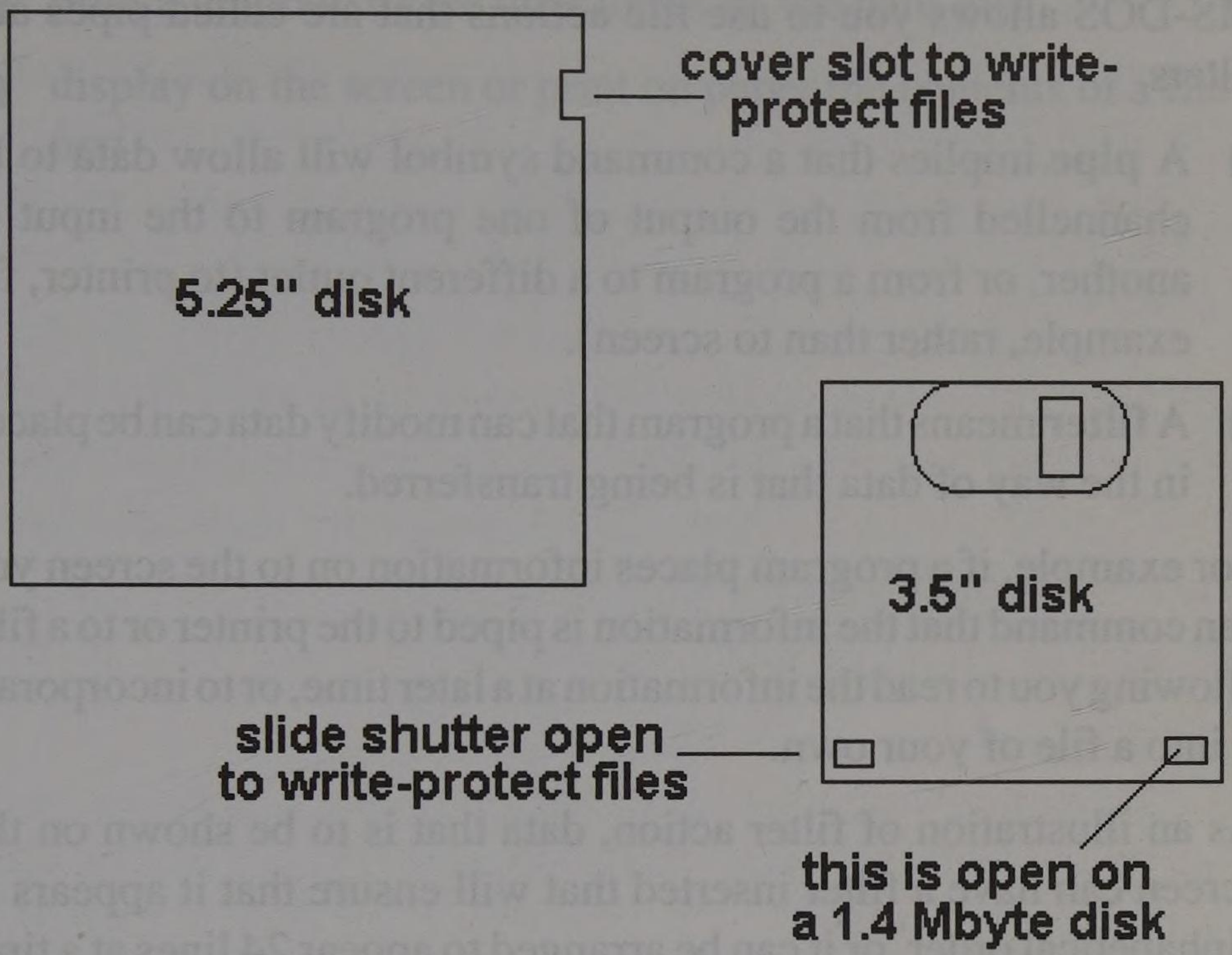
- This applies to the hard disk also, but you very seldom put a new hard drive into a computer. You must **never** attempt to format the hard drive unless it is new and unformatted. Formatting a hard drive that has been used will erase all the data on the disk. You can recover this data if you know how, but only if nothing else has been saved since.

Another option that can accompany formatting a floppy disk is the creation of a **System disk**. A System disk is one that will load in MS-DOS when it is put in the A: drive and the computer switched on.

- This is normally attended to by the hard disk, but it is essential to keep two floppy System disks in case of problems with the hard disk.
- If, for example, the hard disk becomes partly damaged you cannot gain any access to the undamaged files unless you can start MS-DOS running. A System disk allows you to do this.
- Another more recent problem is a **virus** attack. If the hard disk is affected by a virus you can often restore control by starting again using a floppy System disk, but only if the System disk was created before the virus was picked up.

This makes it important to make your System disks as early in the life of your computer as possible, and to store such disks carefully. Once prepared, each System disk must be write-protected, and kept in a cool dry place away from magnets, electric motors and any electrical device that uses magnetism.

A 5.25 inch disk is write-protected by fastening a paper tab around the notch in the edge of the disk. A 3.5 inch disk is write-protected by slipping back the built-in plastic tab so that you can see through the square hole. You cannot write-protect the hard drive (other than by disconnecting it).


cover slot to write-protect files
5.25" disk
3.5" disk
slide shutter open to write-protect files
this is open on a 1.4 Mbyte disk

Write-protecting the two types of floppy disk used on PC machines.

9 Other Actions

MS-DOS allows you to use file actions that are called pipes and filters.

- A **pipe** implies that a command symbol will allow data to be channelled from the output of one program to the input of another, or from a program to a different outlet (to printer, for example, rather than to screen).

- A **filter** means that a program that can modify data can be placed in the way of data that is being transferred.

For example, if a program places information on to the screen you can command that the information is piped to the printer or to a file, allowing you to read the information at a later time, or to incorporate it into a file of your own.

As an illustration of filter action, data that is to be shown on the screen can have a filter inserted that will ensure that it appears in alphabetical order, or it can be arranged to appear 24 lines at a time so that the user has time to look at one screen before seeing the next 24 lines.

Other miscellaneous commands allow you to:

- clear the monitor screen,
- alter the date and time held within the computer
- seek help on any MS-DOS command
- ensure that your computer uses symbols and data appropriate to the country (such as the £ sign, an English keyboard and set of characters, printing dates in day, month, year order, printing numbers with a decimal point and a comma separator for thousands)
- alter the shape of the reminder, called a prompt, that MS-DOS used to indicate that it is ready for a command.

Other Actions 9

- print a diagram of the directories on the hard disk
- display on the screen or print on paper the contents of a file of text

10 MS-DOS Methods

MS-DOS uses some standard methods that you need to become familiar with as soon as possible, and to save repetition these are described here.

When you start MS-DOS 6.0 running you will see on the screen:

C:\>

and this is called the **prompt**. It shows that MS-DOS is ready for a command and it also shows what directory is in use. The symbols **C:** mean that the drive is labelled as C:, which practically always means the hard drive, and the **C:** combination means that the main or root portion of the hard drive is being used, not any other directory.

If the prompt appeared as:

C:\MSDOS>

this would mean that a directory called MSDOS was being used. This directory will be created when you install MS-DOS 6.0 on your computer.

- This can be important, because if you want to run a program you normally need to use the directory in which the program is placed.

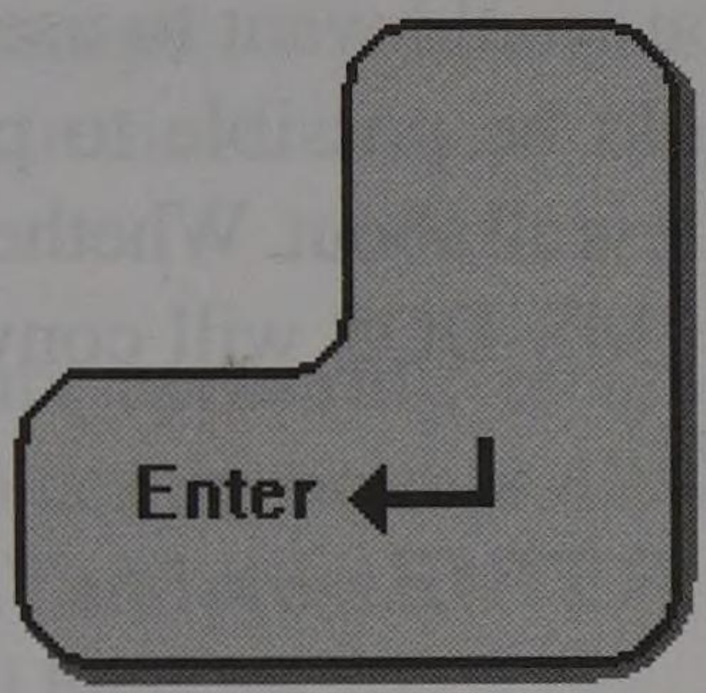
To run a program, you must:

1. Ensure that the correct directory is being used (unless a PATH command has been used, see later).
2. Type the name of the program
3. Press the **ENTER** key.

These steps need to be followed for each program you run, though the internal commands of MS-DOS can be run from any directory, making the first step unnecessary for these commands. In addition, your computer can be set up so that any of the MS-DOS external programs can be run without the need to use the MS-DOS directory directly.

MS-DOS Methods 10

What you type in order to run a program is called the **Command line**, and what is typed in the command line is carried out when the **ENTER** key is pressed. This key is also referred to as the **RETURN** key, but these names very seldom appear on the key, which is placed where the carriage return key (hence the name RETURN) appears on a typewriter keyboard. The key is normally marked with a down-and-left arrow.



The ENTER or RETURN key of the computer is marked with a down and left arrow.

Filenames

You need to know which files are programs and which are not, and this looks like a good point to check out filenames. A suitable filename will consist of up to eight characters, the first of which **must** be a letter. The other characters can be letters, or you can use the digits 0 to 9, or the symbols

$ # & @ ! % ( ) _ { } ‘ ~ ^`

The following characters **must not** be used:

*** + = [ ] ; : , . / ?**

nor can you use:

the **space**

the **tab**

the **Ctrl character**

as any part of a filename.

10 MS-DOS Methods

Unless it's particularly important to you to use the permitted symbols, you should avoid them, particularly symbols like the single inverted quotes, which are easily confused or overlooked. A good rule is to use words with a digit or pair of digits used at the end to express versions, like TEXT001, TEXT002...TEXT015 and so on. Allowing up to eight characters means that you may have to abbreviate names that you would want to use, like ACCREC1 or BOTLEDG1, but it should be possible to provide a name that conveys to you what the file is all about. Whether you use lower-case or upper-case characters, MS-DOS will convert all filenames to upper-case (Capital letters).

Drive Letter and Path 11

The eight-character (or less) main name for a file is essential, but there are two other parts to a file name that are optional.

- One of these is the drive letter, usually A:, B: or C:
- The other is the extension, a set of up to three letters.

The drive letters A: or B: will be used only if the program is on a floppy disk. For example, if you have a program called SPLAT on a floppy disk that is in the A: drive, you can run the program in either of two ways:

1. With the prompt appearing as **C:\>**, type **A:**, and press the **ENTER** key. When the prompt shows as **A:\>**, type the program name of **SPLAT** and press **ENTER**. When the program finishes the prompt will be **A:\>**.
2. With the prompt appearing as **C:\>**, type the command **A:SPLAT** and press the **ENTER** key. When the program finishes the prompt will be **C:\>**

These methods allow you to run a program no matter where it is located, and the second method allows you to return to the hard drive after running a program from a floppy disk. The first method starts by changing to the floppy drive.

This can be extended to using directory names as well as a drive letter. Suppose, for example that the SPLAT program is located in the C:\ODDPROG directory, and you are currently using the C:\ root directory. You can then run the program by typing, as a name,

C:\ODDPROG\SPLAT

and the C:\ODDPROG portion of this full filename is called the directory path.

- This full directory path is required if you are not using, as your current directory, the directory that contains the program you want to run.

The extension is a set of up to three letters that are added to the end of the filename, separated by a dot. Any letters following three will be ignored when the file name is typed. Like the disk drive letter, the extension is optional, and if it is not used, then the dot is not needed.

- This is why the use of the colon or the dot is prohibited within an 8-character main filename.

The purpose of the extension is to convey some extra information about the type of file, though you can make whatever use of it suits your own purposes, within limits. Just as there are forbidden filenames, there are also forbidden extensions, and in particular you should not make use of the extensions **.EXE, .BAK, .HEX, .COM** or **.OBJ** for your files, because these are extensions that are used with special meanings. The list following shows a few 'standard' extension letters and their uses, and you should, if you use extensions at all, either keep to some of these or use some entirely different codes of your own.

BAK A back-up file.

BAS A program that cannot be run unless BASIC has been loaded first.

BAT A batch file of commands to the DOS.

COM A program that can be loaded and run by typing its (main) name.

DOC A text file of documentation for a program.

EXE As for COM, but a longer program.

MSG A text file of instructions.

SYS A file that is used by the operating system.

TMP A file that is created temporarily and wiped later.

$$$ Also a temporary file.

Drive Letter and Path 11

A filename whose extension is COM or EXE is the name of a program file, and the program will run when you type its name - you do not need to type the extension letters. For example, if you are using a directory that contains a program file called WRITIT.EXE then you can run the program by typing:

WRITIT

and pressing the **ENTER** key.

The same applies to files with the COM extension, and the more specialised type with the BAT extension (see later). The BAT type of file is not so much a program file as a file that runs programs.

Some programs require you to add other information, called parameters, on the command line. These parameters are always separated from the program name by at least one space, and they sometimes require you to use characters like the slashmark (/) which must not be confused with the backslash(\).

12 Summary - Running Programs

1. You can change from one drive to another by typing the drive letter followed by a colon and pressing the **ENTER** key.

2. You can run a program that is stored in the current drive or directory by typing the **main** program name and pressing the **ENTER** key.

3. You can run a program that is stored in any other drive or directory by typing the drive letter (along with directory path, see above), then the program name, and pressing the **ENTER** key.

4. You never need to type the extension letters of the program, and you must not type the full-stop that separates the extension from the main name.

Installation

13 Should You Upgrade?

Should you upgrade to MS-DOS 6.0? The answer depends on the type of processor your computer uses. If this is described as 8088, 8086 or even the 80286 processor then the answer might be **no**, because the full benefits of V.6.0 are obtained only for machines using the more advanced 80386 and 80486 chips. Many features of MS-DOS 6.0 are, however, the same as they were in older versions, so that if you are learning to use MS-DOS from scratch the Step-by-Step approach in this book will be useful to you, and you can ignore the parts that are applicable only to V.6.0.

- The reason is that MS-DOS 6.0 contains many utilities that only the more modern machines can use. The modern chips from the 80386 onwards can make better use of added memory than the older chips, and MS-DOS 6.0 contains a utility, MEMMAKER, that will set up the memory for optimum use.

- On the other hand, if you are using an older machine based on the 80286 chip and you are running out of space on your hard disk, you should consider installing MS-DOS 6.0 even if only for its DBLSPACE utility, see Section 62.

If you are using a machine based on the 80386DX, 80386SX, 80486SX or 80486DX chips and using an old MS-DOS version (before MS-DOS 5.0) then you have everything to gain from using MS-DOS 6.0. If you are using a suitable machine and your MS-DOS version is 5.0, then an upgrade is still worthwhile, particularly if you want to use large amounts of memory along with the Windows system. You will also probably want to make use of the DBLSPACE hard disk utility and the other new features of MS-DOS 6.0.

- At the time of writing, computers using a later chip called **Pentium** are becoming available, and MS-DOS 6.0 can be used to advantage on these machines also.

Should You Upgrade? 13

If your computer uses the 5.25 inch low-density (360 Kbyte) floppy drive(s) or even the 3.5 inch low-density (720 Kbyte) drive(s) it is unlikely that you can make effective use of MS-DOS 6.0, because such drives have not normally been used on 80386 or 80486 machines.

- An outstanding feature of MS-DOS 6.0, carried over from MS-DOS5.0, is that it can be easily installed even on a hard disk that is already well packed with files, and the installation no longer puts the existing files in danger though, as always, it is prudent to keep a complete backup of all files.

- In connection with setting up MS-DOS 6.0, and in many other contexts, you will see the term **reboot** used. **Booting** is the action of switching on a machine so that MS-DOS is started running, and rebooting means re-starting the machine. This can be done either:

1. by pressing the **Ctrl, Alt** and **Del** keys together (using the **Del** key on the keypad at the right hand side, not the one in the set of six keys next to the **ENTER** key) or

2. by pressing a **RESET** button on the main casing of the computer, if this button is fitted, or

3. by switching the computer off and, after an interval of about 30 seconds, on again.

- You should be careful about switching a computer off and then on again. If you do not allow a reasonable time between these actions there is a risk of damaging the power supply unit inside the computer.

14 Preparing for Installation

You must start your installation of MS-DOS 6.0 by making certain that valuable files on the hard disk are backed up by normal copies (NOT by using the BACKUP command of older versions of DOS).

- By valuable files I mean files of text, worksheets, graphics, etc., that you have created and which are therefore irreplaceable, not program files that will still be available to you on their original disks.

You should hold backups of all your own files in any case, but it is important to check for backups when you carry out any modification to an operating system. Make sure that you have backups of a tree diagram to remind you of the directory structure, and of the AUTOEXEC.BAT, CONFIG.SYS and any INI files. MicroSoft claims that some actions will survive even a power-cut without loss of data, but it is still prudent to maintain a full backup.

- If all of these actions are new to you, you will need to read ahead to find what you need to do and how to use the COPY command. The important point is to have a copy on floppy disks of everything you need. If installation goes normally you will not need these copies, but it is reassuring to know that they exist. If anything goes wrong, you will be very glad of the copies.]

- A good backup is like insurance - you think very little of it until you really need it.

1. During the course of installation, the old MS-DOS files will be placed on to a floppy disk or disks in the A: drive that will be labelled as **Uninstall** disks. If the A: drive is high density (1.2 Mbyte or 1.44 Mbyte), one blank **formatted** disk will be needed; if the drive uses 360 Kbyte or 720 Kbyte floppies, two blank **formatted** disks will be used.

- You should make sure that, at some stage, you print out and read any README.TXT file, and any other TXT files on the MS-

DOS 6.0 distribution disks, particularly if you use MS-DOS over a network.

- If your computer is part of a network the installation needs to be carried out by the network supervisor, or whoever is responsible for maintaining the network. Network maintenance is only for the knowledgeable and experienced.

2. Lay out, in order, the floppy disks that are used to distribute MS-DOS 6.0. The standard set consists of 7 3.5 inch high-density disks. 5.25 inch high-density disks can be supplied if requested.

Make sure to note any points in the README files that particularly affect you - this applies particularly if you are using a network, upper memory blocks on a 386/486 machine, some types of hardware, or software that is known to present problems. The README.TXT file is likely to change at frequent intervals, but on the early versions of MS-DOS 6.0 the software that was likely to cause problems was either not readily available in the UK or could be upgraded. The problems described in the READ.ME or README.TXT files are therefore not common in the UK.

You can see the READ.ME or README.TXT files on screen by using a command called **TYPE** along with a filter called **MORE** (these terms will be explained later). To see the text of a file called READ.ME, for example, when you are using the directory that contains this file, you type on screen the words:

TYPE READ.ME | MORE

and press the **ENTER** key. The file will be displayed one screen at a time (this is what the **MORE** part does) and you can press the spacebar to move from one screen to the next. You will have to use the command all over again if you want to look at an earlier screen. See later for details of how to print such files.

14 Preparing for Installation

- Remember that the **TYPE** command shown above will have no effect unless you are using the correct drive or directory.

It is important to realise that you cannot install MS-DOS 6.0 simply by copying files to your hard disk, because files are stored in compressed form, and the SETUP utility should normally be used to expand and place these files, though they can be manually expanded and copied if required. The compressed files are marked by using an extension that ends with an underscore, such as MSDOS.SY_ or UNINSTAL.EX_.

- Many actions in MS-DOS 6.0 and its DOSSHELL system are easier to carry out using a mouse, using a single click (press and release) to select and double-click (press and release twice in quick succession) to run or open a file. The left-hand mouse button is used exclusively. If your computer contains no mouse-port, you can fit a serial mouse (which can be remarkably inexpensive if you opt for the Genius mouse) or a Bus mouse that is packed along with an interface card.

- Note that when a pause is used during installation to allow you to read a notice, the **ENTER** key is used to continue installation, the **F1** key to obtain help, and the **F3** key to exit.

You should create a directory to hold MS-DOS utility programs, if one does not already exist. If you do not already have a directory called MSDOS, create one as follows:

Type **C:** (press **ENTER**)

Type **MD MSDOS** (press **ENTER**)

to create this directory. This is very important, because the use of this directory (which should have been created when an older version of MS-DOS was installed) is assumed throughout this book and in many references in other books and in magazines.

NOTE: The following description is based on an installation on a 386 machine that was equipped with Windows 3.1. A machine without Windows would trigger some different messages, and there are also slight differences between monochrome and colour screen machines.

Once you are ready, install MS-DOS 6.0 as follows:

1. Insert the first MS-DOS distribution disk into Drive A:. Switch to this drive by typing **A:** (press **ENTER**), and then type **SETUP** (**ENTER**).

2. The SETUP program will check your system, and remind you to have one or two UNINSTALL disk(s) ready. Remember that you must place these into the A: drive when instructed by a screen message.

3. Wait until any messages about diagnostic reports are completed.

4. The SETUP program will check the complete computer system and report.

■ If some old MS-DOS files are found on the root directory of the hard disk you will be asked to leave SETUP and remove these files. This is an unusual problem, and it needs some experience to cope with - see the *Pocket Book of MS-DOS 6.0* for details. You might require some help from an experienced user to carry out these actions that are required.

5. You are then given the opportunity to continue with SETUP (press **ENTER**), to learn more (press **F1**) or to abandon SETUP (press **F3**).

6. You are reminded of the need to format and label a 3.5" 1.4 Mbyte or 5.25" 1.2 Mbyte disk as an UNINSTALL disk. You will need to prepare two disks if your A: drive (unusually) uses a 360 Kbyte or a 720 Kbyte drive. The UNINSTALL set

contains files that allow you to replace the old version of MS-DOS if any difficulties arise. Press **ENTER** to continue with SETUP.

7. You will see on screen a report on the existing operating system software and video hardware configuration that you can confirm or change as required. This should show the entry for **DOS PATH** (the directory where MS-DOS utilities are stored) as **MSDOS**. If this is not so, type in this name.

■ Throughout this book, we assume that all MS-DOS 6.0 files are stored in a hard-disk directory called **C:\MSDOS**. If this directory does not exist before you install MS-DOS 6.0, you will be asked about creating such a directory, and the name C:\DOS may be suggested. You can either change this to C:\MSDOS by editing the message that appears, or you can accept this name, and remember that wherever **C:\MSDOS** is used in this book, you will need to use **C:\DOS**.

8. You will see on screen the options of **Backup, Undelete** and **Anti-Virus** and you are asked which program options you want to install. By placing the cursor on any of these options you can obtain the further choices of Windows version, MS-DOS version, or both. You will probably want to install all three of these options in one version at least.

■ Note that the disk space required for all three options in their Windows versions is 6,936,128 bytes, considerably more than was needed for MS-DOS 5.0. Note that the options that are presented depend on whether Windows is present in your computer.

■ By using **DBLSPACE** (see Section 62) you can obtain more hard disk space despite this extra disk space requirement.

9. The WINDOWS directory will be shown if Windows is installed - you are asked to confirm (or change in the unlikely chance that the wrong directory is shown).

10. You are reminded that SETUP should not be interrupted from this point onwards, and are given a final chance to leave (press **F3**) or to proceed (press the **Y** key).

11. The process is automatic from this point. You will be asked to change disks at times. The first of these times occurs when you are asked to place the UNINSTALL (or UNINSTALL #1 if there are two) disk into the A: drive and press **ENTER** to notify this.

12. The progress of installation is marked by a display that shows a yellow bar on a blue background. This bar lengthens to indicate the extent of installation. Several messages appear during installation to remind you of facilities that will be available when you have completed installing MS-DOS 6.0. When UNINSTALL is complete, you are asked to return to the first SETUP disk and press **ENTER**.

13. The SETUP program will request you to insert the second disk of the 1.4 Mbyte set after about 33% of installation has been completed. The other disks will be requested in turn. While the disks are being read, you will see messages on the screen about the benefits of MS-DOS 6.0 utility programs.

14. When the last disk has been read, you are asked to remove all floppy disks and press **ENTER**. You are reminded that MS-DOS 6.0 is installed, and that old file versions have been saved on the UNINSTALL disk(s). Pressing **ENTER** again will restart the machine running MS-DOS 6.0 (it will have been running the old DOS up to this point).

15. You should see the screen prompt now appearing as **C:\>**.

16 Making System Disks

It is also important now to make several **SYSTEM** floppy disks from which you can boot the computer if there is at any time a failure of the hard disk. You must make copies also of the files that are called COMMAND.COM, CONFIG.SYS and AUTOEXEC.BAT. You can use new unformatted blank floppy disks or blank formatted disks as you please - the process takes longer if you use unformatted disks. Make at least two such System disks **now**.

1. Place a floppy into the A: drive. Make sure that the prompt on the screen reads **C:\MSDOS>**. If it does not, then type:

   CD C:\MSDOS (press **ENTER**)

2. Type the command:

   FORMAT A: /S (press **ENTER**)

■ Note the space between **T** and **A**, and between the colon and the forward slashmark.

3. You will be asked to insert the floppy and press **ENTER** - since the floppy is already in place, simply press **ENTER**.

4. Wait until the process is complete before you remove the disk - the floppy disk drive light should be out and there should be a message on the screen asking you if you want to format another disk.

5. Insert another disk, press the **Y** key to answer the question about formatting another disk, and then again to acknowledge that you have a disk in the drive. Wait until this disk also is formatted.

6. When the message about formatting another appears, answer with the **N** key. You should see the **C:\MSDOS>** prompt message again.

7. Keep this disk in place and type the following set of instructions, pressing **ENTER** after typing each instruction:

COPY C:\MSDOS\COMMAND.COM A:

COPY C:\AUTOEXEC.BAT A:

COPY C:\CONFIG.SYS A:

- This assumes that COMMAND.COM is in the C:\MSDOS directory and the other two are in the root directory, which is normal. If not, you will have to find where these files are located, and type the correct directory name into the **COPY** line.

8. When the **C:\MSDOS>** prompt returns after the last of these, and the floppy disk drive light goes out, remove the disk and insert the other one, then repeat step 7 above.

9. Now when this disk is ready, remove it from the drive. Label both disks as **MS-DOS 6.0 SYSTEM DISK**, set the write-protect tabs, and store the disks in cold dry place well away from magnets.

These disks are your safeguard in the event of anything happening to the hard disk. You will normally start (boot) the computer with no floppy disks in the drives, allowing the hard drive to provide all the files that are needed to start and control the machine. If anything should go wrong with the hard drive you will get a message that calls for you to place a SYSTEM floppy in the A: drive - and that's when you will be glad that you made these disks.

- A hard drive is a reliable device, but it is a mechanical device and like all mechanical devices will eventually fail. You do not necessarily get any warning of this, which is why it is essential to keep SYSTEM disks and to make backups of all your valuable data.

- System disks are also your front line of assistance in the event of contracting a Virus, see Sections 65 - 69

17 The CONFIG.SYS File

No two computers are completely alike, nor do any two computer owners use the machines in the same ways. Because of this, details of these differences are stored in two files that the computer reads when it is booted up. These files make sure that your computer is set up correctly for such things as the type of keyboard (English, French, German...) and that any programs that you want to keep in the memory are put there. One example of a program you would need to keep in the memory is the driver program for a mouse.

- A program that you load, run, and leave, is normally replaced in the memory when you load another program. MS-DOS needs also to have programs that remain in the memory so as to use hardware like the keyboard and the mouse. These programs are sometimes referred to as **TSR** (Terminate and Stay Resident) because they are not removed from the memory when you stop using them.

The file whose name is **CONFIG.SYS** consists of lines of commands that are written as text (not coded), and it is used whenever the computer is started up.

```
DEVICE=\MSDOS\SETVER.EXE
DEVICE=C:\WINDOWS\HIMEM.SYS
DEVICE=C:\WINDOWS\EMM386.EXE L=3000 RAM
FILES=30
BUFFERS=20
COUNTRY=044,850,\COUNTRY.SYS
SHELL=\MSDOS\COMMAND.COM C:\MSDOS\  /P
DOS=HIGH,UMB
STACKS=9,512
```

A typical CONFIG.SYS file for a 80386 machine. Your CONFIG.SYS will not necessarily be identical because the file controls the machine configuration, which will not be identical for any two machines.

This use of CONFIG.SYS is automatic when the system is started (booted), and the CONFIG.SYS file **cannot** be run like a program on its own.

A simple CONFIG.SYS file for a PC might consist only of the lines:

```
FILES=20
BUFFERS=20
COUNTRY=044
```

and it's likely that you would want to use at least this number of lines, probably many more, and probably with different entries for a different machine, particularly when memory has to be organised on a 386 or 486 type of machine.

- You can alter the CONFIG.SYS file of your computer using the MS-DOS editor (or any other text editor program). Do not use a word-processor unless you know how to use it to create the type of files called ASCII
- Changes that you make in this way to a CONFIG.SYS file have **no effect** until you reboot the PC with the new version of the file.

When you have installed MS-DOS 6.0 on a hard drive, it will provide a CONFIG.SYS file for you. This will usually be satisfactory, because MS-DOS 6.0 can detect the type of monitor system, keyboard, language and other settings that you are using and alter the CONFIG.SYS file to suit.

Sometimes, however, it is not possible for the SETUP of MS-DOS 6.0 to get this information, and your CONFIG.SYS may be incorrect in some details. Your indications of problems are the inability to get the £ sign on screen, some key actions reversed (notably the @ sign and the " sign), dates appearing as 5/30/1993 instead of 30/5/1993, and so on. These indicate that you have not set up CONFIG.SYS for UK use.

To alter this you need to edit the file, using the MS-DOS editor.

- Make certain, before you make any alteration to your CONFIG.SYS file, that you have other copies on floppy disks. See Sections 16 and 25 for using the COPY command to make copy of this file.

The editor program is located in the directory called MSDOS and for the moment the easiest way of working with it is to start by entering (or logging to) that directory by typing:

CD C:\MSDOS (press **ENTER**)

Once the prompt shows **C:\MSDOS>** you can start using EDIT to check your CONFIG.SYS file.

1. Type **EDIT C:\CONFIG.SYS** and press **ENTER**.
2. You will now see your own CONFIG.SYS file displayed. Check that you have the line:

   COUNTRY=044,850,C:\MSDOS\COUNTRY.SYS

   which ensures that you are using the U.K. set of standards.
3. If this line is missing, type it in. You can make space for this line immediately following the line that starts **BUFFERS=20** by placing the marker (the cursor) on the next line and pressing **ENTER**. Do not alter any other lines of the CONFIG.SYS file.
4. When your edited CONFIG.SYS file, save it by moving the mouse so that the cursor is on the **File** menu. Click the mouse button, then move the cursor to the **Save** item and click again.

   Note that we assume that the file called COUNTRY.SYS is located in a directory called MSDOS. Some PC users like to place all files with the SYS extension in the root directory, in which case you would use **COUNTRY.SYS** in place of **C:\MSDOS\COUNTRY.SYS**

The AUTOEXEC.BAT FIle 18

The AUTOEXEC.BAT file is another file that sets up the computer, but in a way that tailors it for what you require, rather than for a specific country or piece of hardware, though the setting for the keyboard is done using a line in AUTOEXEC.BAT. This file exerts its control **after** the computer has loaded up the MS-DOS operating system and after it has run CONFIG.SYS.

- You can run AUTOEXEC.BAT at any time while the computer is running simply by typing **AUTOEXEC** (press **ENTER**). This is not always desirable, but it can be useful.

No two AUTOEXEC.BAT files will be identical, because both computers and computer-users have different requirements. What follows is a list of some typical lines that are likely to be found in the AUTOEXEC.BAT files of machines running MS-DOS 6.0. You are likely to alter your AUTOEXEC.BAT file many times in the lifetime of the computer because the AUTOEXEC.BAT file reflects the way in which you use programs on the machine.

For example, the AUTOEXEC.BAT file on the machine I am using to type this work appears as:

```
@ECHO OFF
PATH C:\MSDOS;C:\;C:\WINDOWS;C:\BATS;C:\EDITOR
SET DIRCMD=/W/P/O:GN
SET TEMP=C:\TEMP
LH /L:1,15904 C:\MSDOS\KEYB
UK,,C:\MSDOS\KEYBOARD.SYS
LH /L:1,6400 C:\MSDOS\DOSKEY/INSERT
LH /L:1,640 KEYCLICK
LH /L:1,56928 C:\MSDOS\MOUSE
LH /L:0;1,42400 /S C:\MSDOS\SMARTDRV.EXE 3072 512
LH C:\MSDOS\UNDELETE /TC-20
REM DOSSHELL
WIN /S
```

and many of the commands in this set are peculiar to this machine and to the way that I work with it. In addition, some lines have been inserted by MS-DOS 6.0 Setup and others by other programs.

- Lines that start with **LH** have been modified by the use of MEMMAKER, see Section 56. For example, the line that was originally typed as **KEYCLICK** has become **LH /L:1,640 KEYCLICK**. This is part of the MEMMAKER action, and you do not need to create the **LH** part of such lines by typing them.

For your own AUTOEXEC.BAT file the most important line is likely to be the PATH line. This determines where the machine will search for batch and program files after trying the current directory. The form of the command is **PATH** followed by a file path and using a semicolon to separate this from the next filepath. For example:

PATH C:\;C:\MSDOS;C:\BATS

ensures that the C:\ root directory is searched first, then the C:\MSDOS directory and then the C:\BATS directory. You can extend this as much as you need, but remember that having a very long PATH line can slow down the action of the computer each time you type a program name without any specific path.

- The importance of this PATH line will be emphasised in later Sections in this book

With **C:\MSDOS** in the PATH line you will never again need to log onto the MSDOS directory to run the Editor, for example, because if you type **EDIT** (**ENTER**) the machine will look for this file and find it in the MSDOS directory.

Keep the PATH statement for **essential** searches. Some programs add to the PATH statement in order to make it easier to start from any directory. It is better to write a batch file to start such programs, specifying the directory, than to allow the PATH line to become bloated in this way.

The AUTOEXEC.BAT File 18

Another line, mentioned earlier, that you are almost certain to need in your AUTOEXEC.BAT file is:

KEYB UK,850,C:\MSDOS\KEYBOARD.SYS

which loads in a memory-resident program, KEYB, that configures the machine to use the UK keyboard (with its £ sign). Some computers make use of the US keyboard and do not need this line- if the line is used with a US keyboard you will find that some keys produce the wrong symbols.

This line assumes that the C:\MSDOS directory contains all of the necessary files, and since MS-DOS 6.0 creates this directory and fills it, the assumption is reasonable unless you are using an earlier version of MS-DOS. The **KEYB** command is a standard DOS command, and the **UK** portion specifies that the UK keyboard is to be used. The number **850** is a code page number and, unless you are using equipment of IBM manufacture, the number is not of much significance. The last portion provides the name of the file, **KEYBOARD.SYS** and its path, and is essential unless KEYBOARD.SYS is placed in the root directory of the hard disk.

If this command is wrongly used you will get an error message **Bad or missing keyboard file** while the machine is loading up. If you do not see the message, incorrect action is indicated by inability to make the Shift-3 key of a UK keyboard produce the £ sign.

MS-DOS 6.0 has a set of commands that were introduced with MS-DOS 5.0 which give much more control over the machine, and in particular, the DIR command is much improved. The older version of DOS would produce a directory listing which filled only half of the screen and which showed files in order of entry only. MS-DOS 6.0 allows the action of DIR to be modified by using an environment variable called DIRCMD, a batch file parameter, whose values are allocated by using the SET command, see Section 45.

- 3 -

Common Commands

19 Common Commands

The most common commands of MS-DOS are available whenever MS-DOS is loaded into the computer; the others are available if:

1. the utility command files are all kept together in a directory, usually named as **MSDOS**, and
2. the PATH statement in the AUTOEXEC.BAT file names the **MSDOS** directory.

so that there should be a line in your AUTOEXEC.BAT file that starts:

PATH=C:\;C:\MSDOS

Remember that you can name this directory when MS-DOS 6.0 is being installed. If you have not done this, or if the directory has used another name you will need to make allowances for this. For example, if you have placed the MS-DOS files in a directory called C:\DOS, this is the name that will appear in the PATH line.

- You might have other names in the PATH line, particularly if you have installed Windows 3.1 any time. The important point is to make sure that the C:\ and C:\MSDOS parts are present.

With this PATH line correct, you can use any MS-DOS command name without the need to specify whereabouts on the hard disk the file is located. Do not be tempted to add too much to a PATH line with the aim of allowing you to have similar easy access to every file on your disk. Each item you add to a PATH line makes searching for files take a longer time, so that the PATH line should specify directories for the most important files only.

The DIR Command 20

DIR is the directory command, the method of finding what directories and files are contained on a disk, or in a directory. You use **DIR** to display all files, details of one file or a group answering to a wildcard description (see later), or to test if a file or set of files is present. You can use the command word **DIR** alone, followed by a drive or path or by a complete file specification, and you can modify its action by adding various option letters. The most general use is to list all files in the current directory, so that the command applies to the current drive, and consists only of **DIR**.

Example: **DIR**

this produces a display, on a floppy disk, such as:

```
Volume in drive A has no label
Volume Serial Number is 375E-13D4
Directory of A:\

LASLAB   DOT        1818   12/03/93   18:58
MACRODE  EXE      105248   09/03/92   12:00
NEWMACRO DOC      110392   08/03/93   10:03
NEWMACRO DOT      110392   08/03/93   10:35
NORMAL   DOT        4800   04/03/93   17:19
WBASIC   CLP       14006   08/03/93   12:41
WBASIC   DOC       20843   08/03/93   12:43

       8 file(s)     367499 bytes
                    1086464 bytes free
```

which shows, in order:

1. The label name of the disk - none in this example.
2. The serial number for the disk.
3. The files that are present on this drive, with the name, size, and the date and time of their creation.

4. A summary of the number of files, the space they take up on the disk, and the amount of space remaining for use.

■ The files will be shown one on each line as a default mainly because the original PC machine used a 40character per line screen display.

When you use the DIR command on a hard disk, you are likely to see something like the following example - this has used a modification of DIR (see later) that bunches the filenames together and omits the other information. This allows you to see the display in a reasonable amount of space.

```
Volume in drive C is IDEDISK
Volume Serial Number is 16CD-1219
Directory of C:\

[123R23]        [ARCHIVE]       [BATS]          [DBLFILE]
[EDITOR]        [MONEY]         [MSDOS]         [STD]
[TEMP]          [UTILS]         [WINDOWS]       [WP51]
AFTCOMP.CHK     AFTCOMP.DIR     AFTSETUP.MSD    AUTOEXEC.BAT
AUTOEXEC.BAK    BEFCOMP.CHK     BEFCOMP.DIR     BEFSETUP.MSD
CHKLIST.MS      COMMAND.COM     CONFIG.SYS      CONFIG.BAK
DRIVER.SYS      DSVXD.386       EMSCOMP.EXE     MONEY.BAT
MOUSE.INI       VMODE.EXE       WINA20.386
    38 file(s)      183630 bytes
                    24526848 bytes free
```

In this DIR display, the directories are marked with brackets, so that each of the bracketed names you see on this display represents a directory that can in turn contain both files and directories.

■ This is the main (root) directory of the hard disk, and it contains mainly directories, very few files. This is as it ought to be, and even this example contains files that you could, with advantage, move elsewhere. The fewer the files in the root directory the

better, because virtually all of your files ought to be gathered into sub-directories.

Fancier DIRs

The modern MS-DOS versions (from 5.0 onwards) allow you to use a much more flexible use of **DIR** with option letters and the use of the auxiliary **DIRCMD** file, see Section 45. You can do this by using an option parameter following the **DIR** command. You need to leave a space following **DIR**, then type a forward slash (/) and then an option letter.

For example, using:

DIR /w

will produce the wide form of display as illustrated above for a hard disk. You can use more than one option, each separated with its slashmark, provided that these do not conflict with each other, and you can also use a minus sign ahead of an option letter to turn off an option that was fixed on.

The full options list is shown in Section 45, and the following examples show the more useful ones. Note that you can use upper or lower case letters:

DIR /b show only filename (no dates or sizes)

DIR /o show the directory sorted in order

DIR /p page the output to one screen at a time

DIR /w display the directory in wide form

If you use the **/o** option to display the directory in sorted order, you can add other option letters to specify how you want the directory sorted. You can use:

DIR /o:n to mean alphabetical sort by name

DIR /o:g to mean that directory names will be sorted ahead of filenames

- If you use option letter **g** along with any others, you must always place the **g** option first.

and the full list is also shown in Section 45. You can also use options together so that **DIR /o:gn**, for example, will produce an alphabetical sort in order of name, with directory names appearing before the list of filenames.

It would be rather tedious if you had to type **DIR /o:gn** each time you wanted to get a directory listing, and it is possible to type a line into the AUTOEXEC.BAT file which will ensure that each time you type **DIR**, the actual command will be the **DIR /o:gn** (or any other enhanced set of options you want). See Section 45 for details.

Wildcards and DIR

A *wildcard* means a character that can represent any other character, and the two wildcard characters that MS-DOS can use are **?** and *****. The **?** sign can represent any single character, and the ***** can represent any set of characters. Using **DIR FIL?**, for example, would produce a list of files with names such as FILA, FILB, FILC, FILD and so on (assuming these files existed on your disk. Using **DIR FIL*** would produce all the examples mentioned, and also names such as FILTON, FILMFUN, FILTER, FILENEW2 and so on.

DIR is very often used with wildcards, but it does not really need to have a wildcard character typed if either the main name or the extension is to be substituted. For example, using **DIR A:FILE** is equivalent to using **DIR A:FILE.*** and it will display all files on the A: drive which have the filename FILE, but with different extensions, such as FILE.DOC, FILE.DAT, FILE.BAT, FILE.GMP and so on. Taking another example, typing **DIR B:.BAT** is equivalent

to typing **DIR B:*.BAT** and it will display any file that has the extension letters of BAT on the B: drive. This would display files with names such as ABBA.BAT, BABBLE.BAT, CALL.BAT, EARS.BAT and so on.

- Though it is not strictly necessary, it is more in line with other file commands to type **DIR *.BAT** or **DIR FILE.***, and the habit is difficult to break.

21 Finding Your Way

When you use **DIR** after starting your computer you will normally see a list of names, some of which are filenames, others being directory names. Using the directory option of a sorted list in the form **DIR /o:gn** allows this list to appear in sorted order, with directories first, and this makes it much easier to find the directory names. In addition, of course, each directory name appears enclosed in square brackets. You now need to know how to find a list of the files in any of the directories.

The important point here is where you are starting from. When you have just started up (booted) the computer it's likely that you will be using the root directory of the hard disk, usually C:\. You might be using another directory, however, so that if you want to get a listing of files in a directory called WRITING, you need to provide the information that MSDOS can use to find where this directory is.

1. One method is to type:

   DIR C:\WRITING /o:gn

   which specifies that you want to list files (and sub-directories) in a directory called WRITING that is located just off the root C:\ directory. After the listing appears, you will be using the same directory as you were using when you started the command.

2. The other method is to use two commands, one to change directory, the other to list files:

   CD C:\WRITING
   DIR /o:gn

   and in this example, you will be using the WRITING directory after the list has been shown.

- In both examples, the options have been shown - remember that you can make these into defaults (always used when you type **DIR**).

Which one you use is up to you, and either way you will see a list of names that will include both filenames and (if they exist) directory names. Directories within directories are sometimes called sub-directories, but in a sense all directories except C:\ are sub-directories in any case.

Working with sub-directories

There are three commands that are concerned with directory use.

1. **CD** (or **CHDIR**) is used to display or to change the directory you are using (the current directory).
2. **MD** (or **MKDIR**) is used to create a new directory that branches off the directory you are using.
3. **RD** (or **RMDIR**) is used to delete a directory that contains no files and which you do not want to use again.

CD used alone (or followed by drive letter only) will display the current directory path that is being used in that drive. **CD** followed by a drive letter can be used to show the current path in a drive that is not being used at present (there must be a disk in the drive). For example, if you are currently using the directory whose path is C:\WRITING\LETTERS, then typing **CD** (press **ENTER**) will produce on screen:

C:\WRITING\LETTERS

as a report. You can also use a line such as **CD A:** to show what path you are using in another drive (the A: floppy disk in this example).

CD is much more commonly used to change from one directory to another, to make another directory the current directory. When you use **CD** in this way, you must follow the command by a space and then the path to the directory you want to use. For example:

CD C:\WRITING\BOOK1

will take the path from the root hard drive to WRITING to BOOK1, leaving this as your current directory.

If you are currently using a directory that is along the path to the one you want, you can omit all the earlier steps. Suppose, for example that you have a set of directories arranged in the path:

C:\WRITING\LETTERS\OLDLETTER\1992_93

and your current directory is LETTERS. To get to the 1992_93 directory you need type only:

CD OLDLETTER\1992_93 (press **ENTER**)

- you do not need to go through the whole path (though there is nothing wrong in doing so). You must go through the whole path if the directory you want is nearer to the root in the same path, or if it is on another path. For example, if you were using the 1992_93 directory as your current directory and wanted to use LETTERS, you must type:

CD C:\WRITING\LETTERS

If you are using LETTERS and you want to use a directory MYSUMS, which is in the path C:\SPREADS\LOTUS\MYSUMS then you have to type:

CD C:\SPREADS\LOTUS\MYSUMS

Using TREE 22

Using **CD** is not difficult - but you have to know how your directories are arranged. Not everyone is so organised as to keep a diagram of each sub-directory position as and when the directories are created, so that the **TREE** utility is a welcome way of finding out how your hard disk is organised.

TREE is an MS-DOS utility, which we assume you have placed in the C:\MDOS directory. We are also assuming that your AUTOEXEC.BAT file contains a PATH line that will ensure that the MSDOS directory will be searched if you type a name of a program file.

The TREE program displays directories and subdirectories in a family-tree form that shows the paths for each. The display is normally on the screen, but it can be placed on paper, which is rather more useful.

- To print the output your printer should be able to cope with the IBM graphics character set - you can arrange for most dot-matrix printers to emulate the IBM ProPrinter, and you can set HP Laserjet or compatibles for the PC-8 characters.

If you use the **TREE** command by itself it will display the tree diagram for the current drive and directory. The alternative is to follow **TREE** by the drive and directory you want to use. Note that using **TREE C:** will result in a diagram that is usually too large to print on one screen or one sheet of paper. You can use **| MORE** following the **TREE** command to force the output into screen pages.

TREE C:\ | MORE

This will give you the output one screenful at a time.

```
C:\>tree a:
Directory PATH listing
Volume Serial Number is 1059-13DF
A:.
    TEXT
        NAMES ——— second layer
            OLD
    GRAPHICS      \ third layer
        OLD
    OUTLINE   \ second layer
```

first layer

root

The form of the screen output from TREE for a floppy disk.

```
Directory PATH listing for Volume HARDDISK
Volume Serial Number is 1A36-730A
C:.
    MSDOS
    OLD_DOS.1
    BATS
    WINDOWS
        SYSTEM
        TEMP
        MSAPPS
            MSGRAPH
            WORDART
            MSDRAW
            EQUATION
            GRPHFLT
    WPWIN
    WPC
    TAP
        DOCS
        TOOLS
    PUBLISHW
        STYLES
        DTP
-- More --
```

One screen page of the TREE diagram for a hard disk.

The first figure shows a typical TREE diagram for a floppy disk with not too many layers of directories, and the second shows the start of a TREE diagram for a hard disk, using the command in the form:

TREE C:\ | MORE

You can make the TREE command provide you with a display of files by using the option letter **/F** - the slashmark is important.

The figure below shows the result of using **TREE A:\ /F** to display the files in each of the directories. The other option, **/A**, is intended for use with old printers. The older Epson printers and several other makes, cannot display the graphics characters that are used to make the lines of a TREE diagram. By using **/A** you force **TREE** to produce its output using ordinary keyboard characters, with dashes and exclamation marks used to make lines.

```
Volume Serial Number is 1059-13DF
A:\
|---TEXT ------------------------- directory name
|       EX2.WSD ----------------- filename
|
|       NAMES
|           FORMLETT.DOC
|
|           OLD
|                   ASKTST.DOC
|
|---GRAPHICS
|       FIG1_1.PCX
|       FIG1_1A.PCX
|       FIG3_1.PCX
|
|       OLD
|               FIG11_3.BMP
|               FIG5_22.BMP
|
|---OUTLINE
        LIST1.WSX
```

The output from TREE A:\ /F showing the files in each directory.

- You should not need to use the **/A** option with any modern printer that you have set to use the PC-8 characters.

23 Creating a Directory

Though a directory name follows the same rules as a filename (it can even have an extension), the computer must create the subdirectory names in the correct form, not as filenames, and to do this requires the use of the **MKDIR** command, abbreviated to **MD**. **MD** must be followed by a path to the directory you want to create - this is normally just the name of the directory.

For example, if you are using (**logged to**) the root directory, and you want to create a subdirectory called WORDS, you do so by typing **MD WORDS** and pressing the **ENTER** key. This will cause some action in the disk drive, and when you use **DIR** from now on you will find the entry:

WORDS <DIR> or **[WORDS]**

depending on the type of directory display you are using, to indicate that WORDS is a subdirectory name, not a filename.

When you subsequently type **DIR \WORDS** you are then requesting the computer to follow the path that starts with the root directory (symbolised by \) and proceed to the WORDS subdirectory. This will give the display of the type given below, showing that there is just one file stored in this particular subdirectory, and the only other entries are the dot and doubledot. Note that MS-DOS always counts this set as being **three** files. Even in an empty sub-directory, the dot and double-dot are always counted as files.

```
C:\>dir \words

 Volume in drive C is HARDDISK
 Volume Serial Number is 1401-311C
 Directory of C:\WORDS

[.]                 [..]                FIG5_9.TXT
        3 file(s)          1686 bytes
                      102973440 bytes free
```

A sub-directory with one file stored - but note that 3 files are reported.

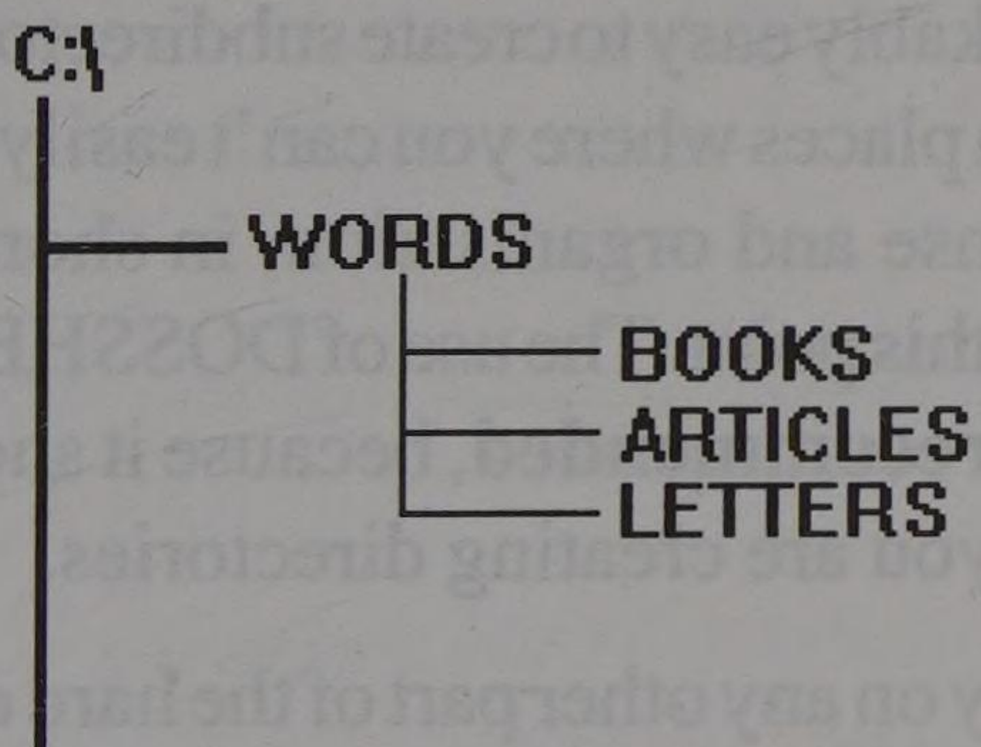
The dot is an abbreviation for the current subdirectory, **[Words]** in this example, the **child** directory

- The double dot indicates the previous directory in the path, the parent directory which in this example is the root directory.
- You can use these dots as if they were valid directory names in commands that require a directory name - you can, for example, use the command:

  COPY ..\myfile.doc .

  with a dot placed at the end of the command, separated by a space from the name, to copy a file from the parent directory to the current directory. This form of command requires very little typing, and is not used to a great extent, mainly because it is difficult to distinguish one dot from another. One point to note is that any parent-child pair can use the same dot and double-dot form of command, so that the directory name is not needed.

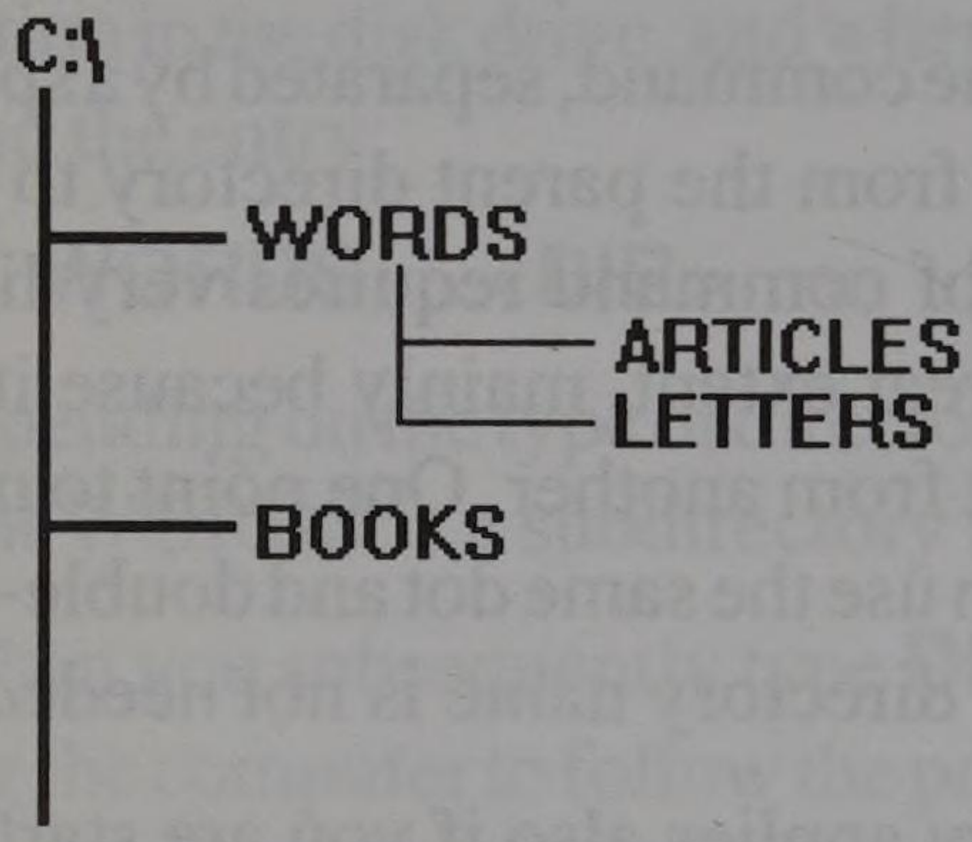
Using **MD** to create a subdirectory applies also if you are starting from another subdirectory. If you switch to the WORDS subdirectory, for example, you can create the further subdirectories of BOOKS, ARTICLES and LETTERS. Each of these, will be a child directory of the WORDS directory.



The three new sub-directories of WORDS illustrated. Each is a *child* of WORDS.

23 Creating a Directory

You have to be in the WORDS directory in order to create these subdirectories of WORDS. If you are in the root directory when you use **MD BOOKS**, for example, you will certainly have created a subdirectory called BOOKS but it will be a subdirectory of the root, at the same level as WORDS rather than as a subdirectory of WORDS. This implies that if you are setting up a hard disk from scratch by transferring a large number of files from floppy disks it is a considerable advantage to plan carefully first, using a tree diagram of the type that has been illustrated.



An incorrect sub-directory location caused by using the MD command from the wrong place (the root directory in this example).

Without such a diagram, it is remarkably easy to create subdirectories that you don't need, and put files in places where you can't easily use them. A spoonful of common sense and organisation, in short, is worth a bucketful of technology at this point. The use of DOSSHELL (see Sections 30 - 43) is **strongly recommended**, because it shows you the directory tree display as you are creating directories.

You can also create a new directory on any other part of the hard disk or on a floppy. For example, using:

MD C:\WORDS\LETTERS\NEWLET

will create a sub-directory called NEWLET if the directories of WORDS and LETTERS already exist, and the LETTERS directory has no existing sub-directory called NEWLET

You can use a sub-directory name more than once. For example you can have a subdirectory called GATHER whose path is C:\WORDS\ARTICLE\GATHER and another one whose path is C:\WORDS\BOOK1\GATHER. These will be totally separate subdirectories as far as the computer is concerned (whether you can avoid confusion is another matter)

24 Removing a Directory

The most difficult part of using a directory tree consists of closing down a directory that you no longer need. You might want to do this because you have found a better way to organise your directories, or because the programs in the directory have been superseded by others.

- You cannot delete a root directory in any circumstances.
- You cannot delete the subdirectory that you are currently using,
- You cannot delete by normal methods a directory that contains files.
- You cannot delete by normal methods a directory that contains a sub-directory.
- There is a specialised command **DELTREE** that allows you to delete directories that contain files, but you should not use it until you have considerable experience.

It is always desirable to start any deletion operations from the root directory. If, taking the conventional example, you wanted to close a C:\WORDS\ BOOKS subdirectory that contained files with the TXT extension (but no further sub-directories), you would need to continue as follows:

1. Use **CD C:** to get to the root directory .
2. Use **DEL WORDS\BOOKS*.TXT** to delete all of the files that use the TXT extension - note the use of a wildcard.
3. Use **RD WORDS\BOOKS** to delete the BOOKS sub-directory.

- Step 2 is something to be careful about. When you delete a set of files using a wildcard, you should always select the files by some common factor, such as the TXT extension in this example. If you use **DEL *.*** you will delete every file in a directory. This

is fine if you intend to delete every file, but it is a disastrous command if you are currently using the root directory. You are always prompted to reconsider the **DEL *.*** command, and you should always do so. See Section 60 for un-deleting an unintended deletion.

You cannot delete the subdirectory, using **RD BOOKS**, until you have first deleted all the files in the subdirectory with a **DEL BOOKS/*.*** command**. RD** is short for **RMDIR** (remove directory), and it cannot, remember, be applied to the directory you are presently using. In other words, you cannot remove subdirectory BOOKS if you are working in this directory, you must work either from the root directory or from another subdirectory that is not a child (or any other descendent) of BOOKS, in this example.

The requirement to empty the directory of all files before deleting it can sometimes be a nuisance, though it does offer a considerable safeguard against deleting valuable files by making you consider all the files that have to be deleted. You can, however, shortcircuit all this work by using the **DELTREE** command which will clear out a directory and delete it in one command. This utility should be used very carefully because it can be very destructive. My favourite nightmare is of using **DELTREE** when using the C:\root directory.

A directory can also be renamed, and its contents can be moved, for details see Section 29.

25 The COPY Command

The **COPY** command is the most-used of all MS-DOS commands because copying is one of the most frequently needed actions. It can:

- copy files to from one disk or directory to another,
- copy text files to the screen.
- copy prepared text files to the printer,
- alter the date and time of files,
- combine several text files into one single copy.

The most fundamental and the simplest action is copying one disk file to another disk or directory. You can do this either retaining the name, or with renaming. The form of the command is

COPY *source destination*

with the files ***source*** and ***destination*** specified. The words must be separated by at least one space.

Example: **COPY C:\WORDS\myfile.txt A:**

will copy a file called MYFILE.TXT in the C:\WORDS directory so that it is placed on a disk in the A: drive, using the same name of MYFILE.TXT.

Points to watch are:

1. There must be a space between the command and the source, and between the source and the destination.
2. If you do not specify a destination filename, the current directory and drive is assumed; but this must **not** be where the source file is located you cannot copy a file to its own drive and directory.
3. If only a drive letter or path is shown as the destination, the file is copied to this drive or directory using the same name.
4. You can use a wildcard in the name, and the first file to fit the description will be copied. If more than one file fits and no renaming is used, all files that fit the wildcard will be copied.

Example: The C:\TEXT directory is being used when the command:

COPY A:oldata.txt

is used. This will copy the file OLDATA.TXT from the A: drive to the C:\TEXT directory, using the same filename of OLDATA.TXT. This file will now be on the hard drive as C:\TEXT\OLDATA.TXT.

Example: **COPY A:OLDWORK.DTA C:\OLDSTUFF**

will copy the file called OLDWORK.DTA on the disk in the A: drive into the OLDSTUFF subdirectory on drive C:, using the same name.

Example: **COPY B:*.TXT C:\TEXTFILE**

will copy all the files with extension TXT on the disk in drive B to the directory TEXTFILE in drive C.

- If you type both a destination drive (or path) **and** a filename, the file will be copied and renamed.

Example:

COPY A:OLDTEXT.DOC C:\REWORK\NEWONE.DOC

will copy the file called OLDTEXT.DOC from drive A to the subdirectory REWORK on the C: drive, and will rename this file as NEWONE.DOC.

- You can use wildcards to copy and change the names of a number of similar files.

Example **COPY A:OLD?.* C:\NEWSET\NEW?.***

will copy files such as OLD1.TXT, OLD2.BAT, OLD3.DOC and so on to the files NEW1.TXT, NEW2.BAT, NEW3.DOC, in which the new files take the same value for the wildcard characters as applied to each of the old files. All the copied files are in the C:\NEWSET directory.

When these copies are made, the directory entry for the copied file shows the **original** date and time of creation. You can alter the date and time as part of the copying process by adding **+,,** to the **source filename** in the copy command, except when the copy is being made to the current drive, when only the **+** is needed. This form of the **COPY** command cannot usefully employ wildcards, because only the date and time of the first file that is copied will be changed.

Example: **COPY A:OLDFILE.DOC+,, C:\NEWSTUFF**

will copy OLDFILE.DOC into the subdirectory NEWSTUFF, and change the date and time information to the date and time of copying rather then the time and date of the original creation of the file.

Example: **COPY A:DATAFIL.TXT+**

will copy the file DATAFIL.TXT from the A: drive to the current drive and will change the date and time to the date and time of copying.

Copying to and from Devices 26

A *device*, in this context, means the screen, a serial port, or a printer, and MSDOS allows another device called **NUL** that will simply do nothing with a file. NUL is a way of preventing a message from being seen in some circumstances, or for testing that a COPY command will work, without the need to wait for anything to be copied. All devices are referred to in the command by abbreviations, which are:

CON for the monitor screen **and** keyboard.

PRN or **LPT1** for the main printer

NUL for the Nul device.

Note: You can use the letters **AUX** or **COM1** for the serial port, but **COPY** will not necessarily receive **from** this device, although it can send ASCII text (no Crtl characters) **to** the serial port. The printer can use **LPT1** or **LPT2** if two parallel ports are fitted.

You would use **COPY** with device names normally to send a file to the screen or printer, but you can also use **COPY** to duplicate text from the keyboard to a disk file or directly to the printer. When you create a disk file in this way, the copy action runs until you press the key combinations **Ctrl-C** or **Ctrl-Break**. When you use copying from keyboard to printer, the end of each line must be signalled with **Ctrl-N** before the **ENTER** key is pressed on the line, and the printing is done when **Ctrl-Z** is pressed, and then **ENTER**.

Example: **COPY A:LETTER1.DOC PRN**

will copy the file LETTER1.DOC from a disk in the A: drive to the printer. You can use a wildcard in the filename if you are not sure of the precise name. The file **must** be in ASCII code, and this means that files made by many well-known word-processors will produce strange effects, depending on the type of printer, because of the use of codes in the range 128 to 255, or in the range 1 to 31. Most wordprocessors have an option that allows you to record files in

ASCII form, and another solution is to pipe the file through a filter program (if you can find one) that will correct the characters into ASCII codes.

Example: **COPY C:\OLDTEXT\LET5.DOC CON**

will copy the file LET5.DOC, in the OLDTEXT subdirectory, to the screen. You can use a wildcard in the filename.

Example: **COPY CON C:\BOLDPRNT.BAT**

will copy whatever you type on the keyboard into a file called BOLDPRNT.BAT until you press the **Ctrl-C** (or **Ctrl-Break**) keys.

Example: **COPY CON PRN**

will copy what you type on the keyboard to the printer when you finally press **Ctrl-Z** and then **ENTER**. The end of a printed line must be marked with **Ctrl-N**.

Combining files

The source files are shown following the **COPY** command word in order, using the **+** sign between each pair. The destination file is shown in the usual way. You can take source files from any disk or directory provided that no disk changing is needed during the copy action (though it **is** possible to swap the disk in one drive while a file is being copied from another drive).

Example:

COPY A:FILE1.DOC+C:\OLD\FILE2.TXT C:\NEW\NEWONE.TXT

will copy FILE1.DOC from the A: drive, and then append the file FILE2.TXT from the OLD directory, naming the new file NEWONE.TXT and placing it in the NEW directory. Note that there **must be no space** on either side of the **+** sign.

Copying to and from Devices 26

This form of **COPY** will also take a wildcard description, so that you can use it to join a number of files with a very compact command.

Example: **COPY A: NOTE?.* B:TOTAL.TXT**

can be used to join files in the A: drive, called NOTE1.TXT, NOTE2.DOC, NOTE3.WRD and NOTE4.ADD into one file called TOTAL.TXT on the B: drive.

27 COPY Options

The **COPY** command allows three option letters **A**, **B** and **V** to be used. Of these, **V** is the most useful, as it checks that the copy has been made, with an error report if it has not, or if the copy is not of the same size as the original. This is equivalent in action to the **VERIFY** command, and its use will slow down COPY actions. Using the **/V** option does **not** check that the copy is identical to the original; you need to use **FC** for that purpose.

Example: **COPY A:PRECIOUS.TXT C:\VALUE/V**

will copy the file called PRECIOUS from the A drive to the subdirectory called VALUE on the C drive, and verify that the copy has been made.

The **A** and **B** options are intended to specify **A**SCII and **B**inary files respectively. The difference is that an ASCII file of text has its end marked with the Ctrl-Z character, code 26, but a binary file must always be copied until the last byte, as indicated by the length-of-file number, has been dealt with. If, as is normal, you do not use these options, COPY assumes a binary action for any straightforward copying of files, and an ASCII action when files are being combined. You can expect trouble, however, if you are trying to make an ASCII copy of a word-processor file that uses the Ctrl-Z characters for purposes other than marking the end of the file. For this reason, **COPY** with no options will use binary copying as a default so that it is normally un-necessary to specify **/B** unless you previously specified **/A**.

- Use **/A** for a **COPY** to a device (such as a printer) which might carry out specialised actions (like changing print size) when it received a non-ASCII character.

Example: **COPY A:TXTFIL.DOC/A B:**

will copy TXTFIL.DOC from drive A to drive B, copying everything in TXTFIL up to the end-of-file byte, code 26. This byte is **not**

copied to the B drive, so that the file copy will **not** be identical. Subsequent uses of **COPY** will be ASCII copies unless you have specified the **/B** option.

Example: **COPY A:TXTFIL.DOC/A B:/A**

will copy TXTFIL.DOC from drive A to drive B, and will copy the end-of-file byte also. Subsequent copies will be ASCII copies unless the **/B** option is specified.

Example: **COPY A:PROGRAM.COM/B B:**

will copy the binary (program) file PROGRAM.COM from drive A to drive B, treating this as a binary file. The **/B** option is needed only if a previous use of **COPY** employed the **/A** option. There will be an end-of-file byte added to the copy, so that the copy is not identical to the original.

Example: **COPY A:PROG.EXE/B B:/B**

will copy the binary file PROG.EXE from drive A to drive B, with no end of file marker on the copy. This is required only if a previous **COPY** command used ASCII options on both files. This is a better form of option for copying files that are liable to contain Ctrl-Z characters inside the text.

- You very seldom need to use these options, because the default action of **COPY** is usually all that you need. This set of actions is included only to show what might be needed if a straightforward **COPY** action is not suitable.

28 Renaming Files

The MS-DOS 6.0 command **REN** (or **RENAME**) is used to rename a file, or a set of files. You might use this because:

1. you want the files, which currently have no extensions or use different extensions, to carry a standard extension such as TXT
2. you need to impose some common pattern on the names, such as changing FIRST, TWO and THREE into CHAP1, CHAP2 and CHAP3.
3. you need to make use of a backup file called, for example, MYTEXT.BAK, but this cannot be read by your word-processor until it has been renamed MYTEXT.DOC
4. you need to alter a sequence, so that files called FIG1, FIG2, FIG3 should be renamed as FIG2, Fig3, Fig4

Manuals show the **REN** command as having the form:

REN *oldname newname*

meaning that you type the name **REN**, a space, the name of the file as it exists at the moment, another space and then the new name you want to use. In its simplest form this presents no problems, so that the command:

REN myfile.TXT oldfile.DOC

will change the name of MYFILE.TXT to OLDFILE.DOC. You will no longer find a directory entry for MYFILE.TXT

- As you might expect, you cannot carry out a renaming if there is already a file in the same directory that carries the name you want to use. In the above example, there must not be a file already existing with the name OLDFILE.DOC. The filename that you want to use must be valid by the rules of MS-DOS.

The simple example assumes that you are using the directory that holds MYFILE.TXT, but you can use **REN** to work on a file in a

different directory. For example:

REN C:\TEXTFILE\MYFILE.TXT OLDFILE.DOC

will change the name of a file that is in the specified directory.

- You cannot, however, type a name for a directory in which you want the renamed file placed, because **REN** cannot change directories. The second name that you type must be a filename alone, with no drive letter or directory path.

The use of **REN** becomes more complicated and more useful when you use it with wildcards, but you need to know what you are attempting. You can use either the **?** (single character) or the * (multiple character) wildcard.

1. You can use **REN *.OLD *.NEW** to change files such as MYFILE.OLD and PIXFIL.OLD into MYFILE.NEW and PIXFIL.NEW

2. You can use **REN CHAP?.TXT CHAP?.DOC** to change CHAP1.TXT, CHAP2.TXT and so on into CHAP1.DOC, CHAP2.DOC (but not MYFILE.TXT into MYFILE.DOC)

3. You can use **REN CHAP?.* CHAP?.DOC** to change files such as CHAP1.OLD, CHAP2.TXT and CHAP3.NEW into CHAP1.DOC, CHAP2.DOC and CHAP3.DOC. This would be impossible if you had more than one file answering any description, for example, if you had both CHAP1.OLD and CHAP1.TXT.

What you need to be careful about is a renaming that is impossible, like *.* to ***.DOC.** You will get a warning message, usually:

Duplicate file name or file not found

when you attempt an impossible renaming like this. A useful rule for renaming is that you should use only one asterisk wild card in a name, so that you can use a wildcard for the main name or the extension

but never for both. Another thing to watch is trying to rename all of a set of files to the same name, such as changing *.TXT to MYFILE.DOC. In this example, **REN** will work on the first file and then refuse to convert any others because the file MYFILE.DOC now exists and nothing else can use this name.

Deleting and Moving Files 29

Deleting a file uses the **DEL** command, with a space separating this from the name of the file that you want to delete. You can use this along with wildcards, but you must be careful to avoid deleting files that you might have preferred to keep.

- Files can be undeleted if you act in time, see Section 60

You can use **DEL** in forms such as:

1. **DEL C:\TEXT\MYFILE.TXT** to delete this named file in this directory. This will not affect any file of the same name in another directory.
2. **DEL C:\FOLIOS\CHAP?.TXT** to delete files such as CHAP1.TXT, CHAP2.TXT, all the way to CHAP9.TXT, but not CHAP10.TXT or higher numbers. Once again, only files in this stated directory will be affected.
3. **DEL C:\NEWWORK*.DOC** will delete all files that use the extension of DOC and are contained in this directory. This use of **DEL** is more risky, because if the directory contained a large number of files you might not notice a few that you would have preferred to keep.
4. **DEL C:\ALLWORK*.*** will delete all of the files in this directory, regardless of main name or extension. This is one of the most risky uses of **DEL**, and when you use this form you will be asked to confirm that you really want to delete all the files in the directory.

- Never use **DEL** with wildcards when you are working with the root directory on the hard disk. You could wipe out most of the files you need, including AUTOEXEC.BAT and CONFIG.SYS, though the essential MS-DOS 6.0 files will not be deleted.

29 Deleting and Moving Files

DELTREE

MS-DOS 6.0 contains a command **DELTREE** that was not available in older versions. This is a command that you should use only with great care because it can not only delete all the files in a directory, it will also delete all subdirectories. For example, if you have a path:

C:\MAINTXT\BOOKS

and you use **CD C:\MAINTXT\BOOKS** to move to the BOOKS directory, you might find that using **DIR** showed that there were directories called BOOK1, BOOK2, BOOK3 and so on, each with files, and that BOOKS itself also contained some files.

Using **DELTREE** with the BOOKS directory as current directory will wipe out all the files in the BOOKS directory, and it will wipe out all the files in BOOK1, BOOK2, BOOK3 and all the other subdirectories. It will then remove these subdirectories, so that all you are left with is an empty BOOKS directory.

- This is a command that you must use sparingly, and only after you are convinced that you will not need the files or the subdirectories again. Do not rely on being able to recover files after using **DELTREE**.

MOVE

The **MOVE** command is also new in MS-DOS 6.0, and it carries out the actions of copying and deleting. You can use it in much the same way as **COPY**, with a source name and a destination name, but without any wildcard characters. When a **single file** is moved, it can also be renamed in the process. If the destination directory contains a file with the same name, this file will be over-written by the moved file.

Deleting and Moving Files 29

You can use **MOVE** also to rename a directory, using the old name as the FROM part and the new name as the TO part of the command. The restriction is that both the old and the new names must themselves be within the same directory - you cannot use **MOVE C:\OLDNAME C:\TEMP\NEWNAME** for example.

Example:

MOVE CHAP1.TXT CHAP2.TXT C:\BOOKS\BOOK1

DOSSHELL

- 4 -

DOSSHELL

30 Introduction to DOSSHELL

The **DOSSHELL** is a program that allows you to carry out MS-DOS actions by using the mouse and executing commands by pointing and clicking the mouse button. The mouse methods that you can use with the DOSSHELL are called click, double-click and drag.

1. **Clicking** means selecting an item that you want by placing the cursor over the item name or icon (picture) and depressing the left-hand mouse button and releasing it, fairly quickly. If the mouse arrow is over a filename at the time, the name will change colour (or shading) to show that it has been selected for one of the actions of DOSSHELL. If the cursor is over the name of a menu (top line of screen) the menu will be displayed.

2. **Double-clicking** means pressing and releasing the left-hand mouse button twice in quick succession. If the mouse arrow is over a program name at this time, the program will run. If the cursor is over an item on a menu, that item will be chosen and used by any action that you select from the **Files** menu.

3. **Dragging** means moving the mouse while an item is selected and the left-hand button is being held down. This is a way of making the scroll-bars (see later) on the directory or files boxes move on the screen when you move the mouse along your desk. The scroll-bars are released when the mouse button is released. Dragging is also used on file moving and copying actions.

You can start DOSSHELL from the now familiar DOS command line **C:\>**. You do this by typing the command word **DOSSHELL** (press **ENTER**), assuming that you have placed the DOSSHELL files in a directory called C:\MSDOS and that the AUTOEXEC.BAT file contains a line that starts:

PATH=C:\;C:\MSDOS

- you can name other directories following this, but C:\ and

C:\MSDOS must be present.

The alternative to starting DOSSHELL manually is to start it automatically each time you switch on the computer. To ensure this:

1. Start the Editor and specify C:\AUTOEXEC.BAT as the file to open.
2. Move the cursor to the end of the file.
3. Type as the last line:

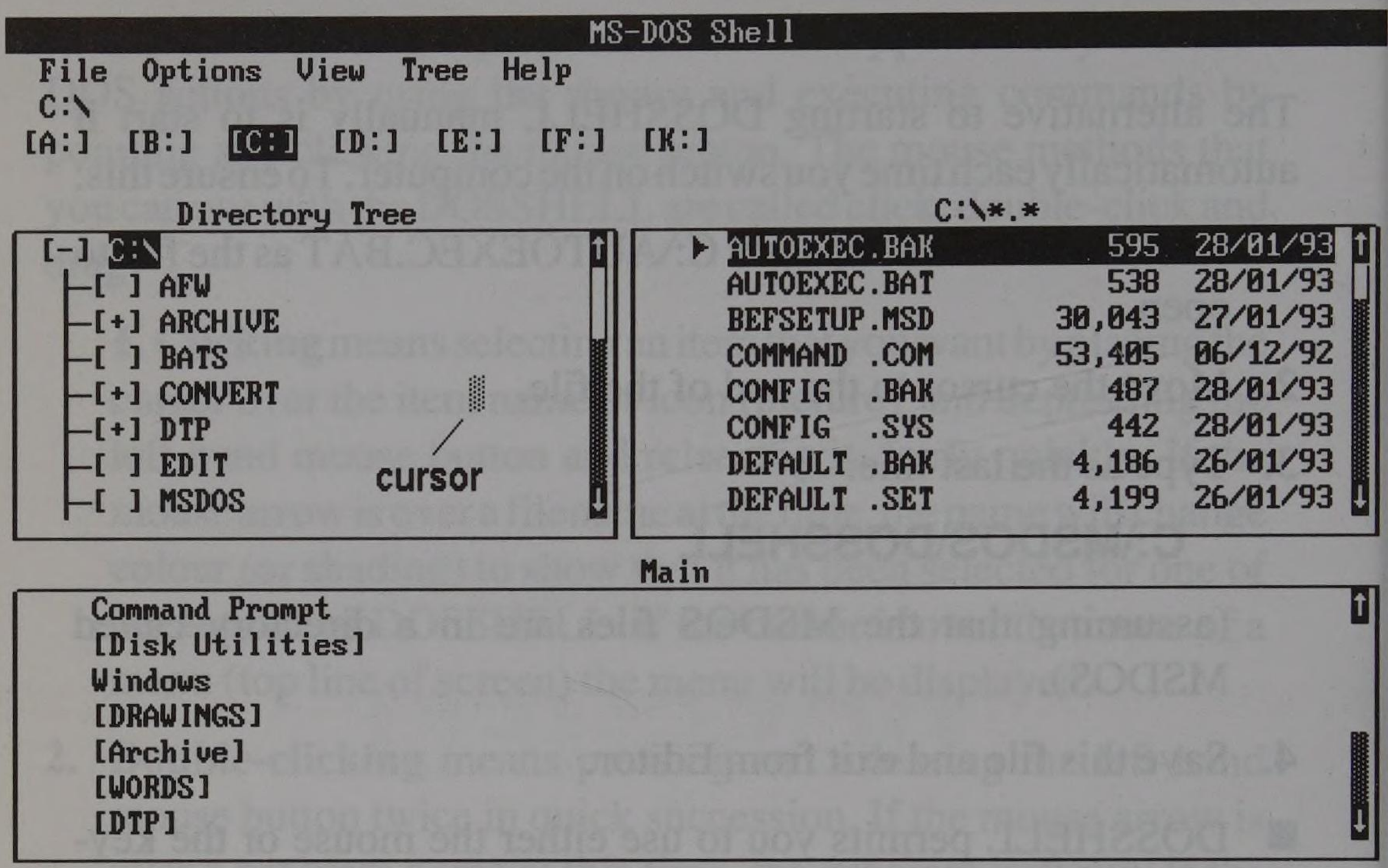
   C:\MSDOS\DOSSHELL

   (assuming that the MSDOS files are in a directory called MSDOS).
4. Save this file and exit from Editor.

■ DOSSHELL permits you to use either the mouse or the keyboard for control. Using the keyboard is slower and tedious, and this book emphasises mouse methods. This choice, however, means that every action can be carried out in at least two ways. Methods that are not described here are either too advanced for this book, or they are an alternative way of carrying out an action that has already been described using a simpler method.

DOSSHELL, as you first see it, is illustrated overleaf. It consists of three sections with a set of option and drive selections at the top of the screen. There are reminders about key actions at the foot of the screen. The directory **Tree** window shows the layout of directories on the disk, with a **+** sign used to indicate a directory that contains subdirectories. The window on the right hand side of the upper part of the screen shows the files in the marked directory - the default starting point, as shown here, is the root directory.

30 Introduction to DOSSHELL



The initial appearance of DOSSHELL when it is first used on a text screen - this appearance will change when you tailor DOSSHELL for your own use.

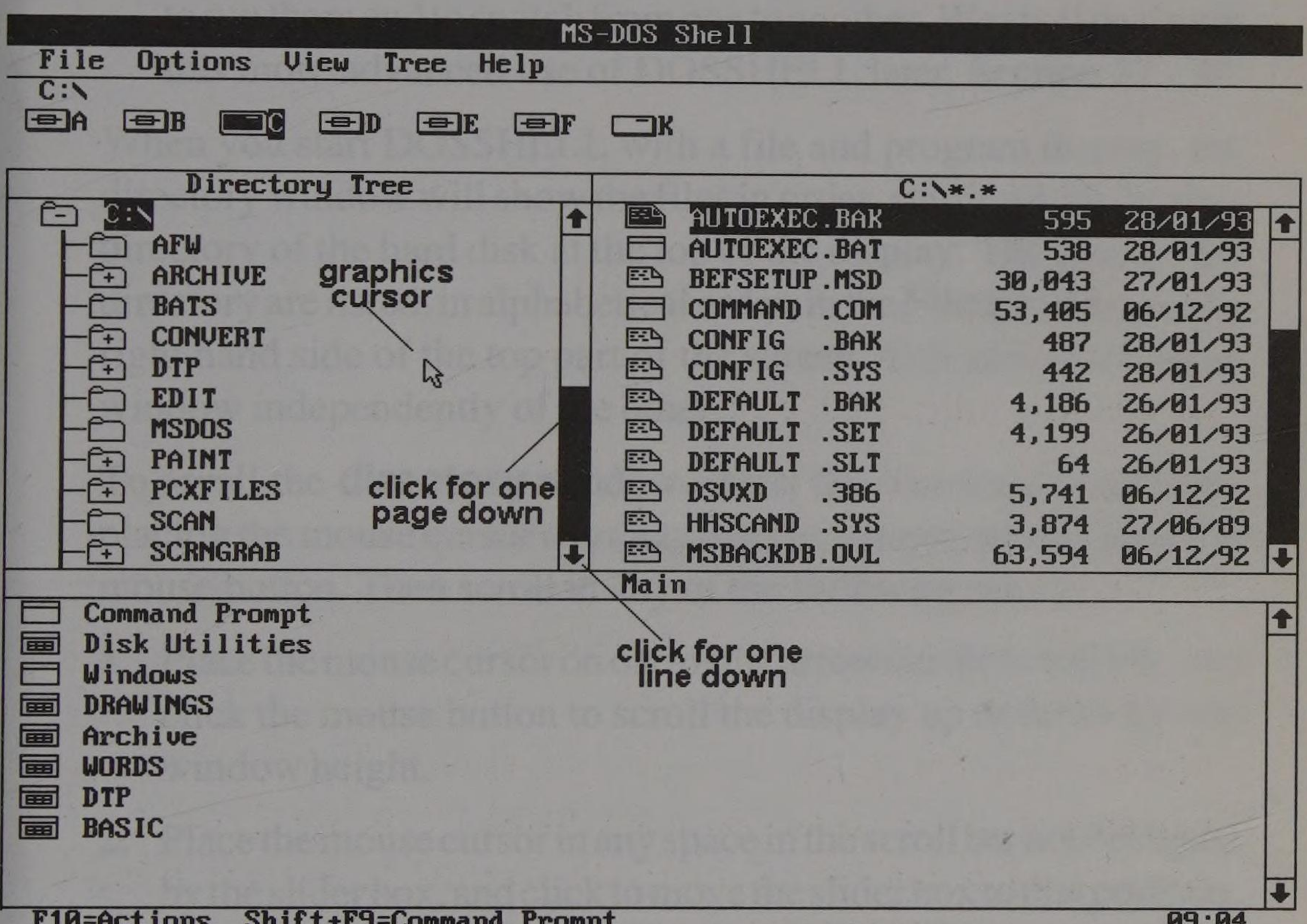
- Each time you start DOSSHELL running it will appear the way it was when you last left it - any changes you make will appear when you run the next time. This is because the DOSSHELL settings are stored in a file (called DOSSHELL.INI) and read each time DOSSHELL is started.

The movement of the mouse controls the movement of a cursor that will appear on the screen when DOSSHELL is in use. The default is to use the familiar 80-character 25 line screen, the text screen, with the mouse cursor shaped like a shaded (or coloured) box.

- When you place the mouse pointer on to a directory and click the mouse button the files list of DOSSHELL will show the files of

that directory. This is the DOSSHELL equivalent of using **CD *directory*** followed by **DIR**. The action is described as **clicking on the directory**.

The mouse is also be used to control the scroll bars. These are the sets of lines at the right-hand side or bottom of a display window. Dragging the button of a scroll-bar will shift the text or the picture in a window. Drag the scroll-bar up to move the image down, and drag the scroll-bar down to move the image up. Try this for yourself - click on the name of the MSDOS directory, and then move the cursor to the scrollbar in the Files window and use the scrollbar to move up and down the files list.



The scroll-bar on a DOSSHELL screen which can be used along with the mouse to display different parts of the directory tree. The Graphics option is in use.

- Clicking below the level of the little box on the scrollbar will move the text up by one window (by the number of lines in a window)
- Clicking above the level of the box will move the text down by one window
- Clicking on the arrowhead at either end of a scrollbar will move the text up or down by one line.

The File and Program Option 31

The file and program option, which is the default when you first make use of DOSSHELL, shows directory and file information in the upper part of the screen. The program information shown on the lower half of the screen is **not** of the program files that exist on the disk, but program **groups** (see later) that you can arrange to display in this portion. Some groups already exist when you initially run DOSSHELL.

- The idea of groups is a comparatively new one in MS-DOS, introduced with MS-DOS 5.0 (though it has been used in MS Windows-3 and other shell programs). The principle is that programs that belong together can be grouped to make it easier to run them and to switch from one to another. We shall deal with this more advanced use of DOSSHELL later, Section 37 - 39.

When you start DOSSHELL with a file and program display, the directory window will show the files in order, starting with the root directory of the hard disk at the top of the display. The files in this directory are listed, in alphabetical order, in the **Files** window at the right hand side of the top part of the screen. You can scroll either window independently of the other.

To scroll the **directory** window, select the directory window by placing the mouse cursor over any directory name and clicking the mouse button. Then scroll in any of the following ways:

1. Place the mouse cursor on one of the arrows in the scroll bar, and click the mouse button to scroll the display up or down by one window height.

2. Place the mouse cursor in any space in the scroll bar not occupied by the slider box, and click to move the slider box to that position.

3. Place the mouse cursor over the slider box in the scroll bar and drag the box to a new position. Release the mouse button when you have dragged the box to its new place.

31 The File and Program Option

If you want to scroll down the **filenames** in a single directory, you must first select that directory by clicking on its name. You then shift to the files window by moving the mouse cursor to that window and clicking. You can then use the same scroll-bar actions as described for the directory window.

- You must make certain that you are using the correct window before you attempt any scrolling actions. The files display will not alter when you scroll the directory, and the directory display does not alter when you scroll the files window. If you select a directory name by placing the mouse cursor on the name and clicking, the **Files** window will alter to show the files in that directory.

The drive letter selection

The opening DOSSHELL display will show the available drive letters in a space near the top of the screen. DOSSHELL will list each drive as a letter enclosed in square brackets, such as **[A] [B] [C]**. The default will be **[C]** when you are using a hard disk. A machine that runs on a network, or uses INTERLNK (see Section 70) or DBLSPACE (See Section 62) will show a large number of drive letters. Some of these may not be real drives.

- You can change to another drive simply by placing the mouse cursor on another drive letter and clicking the mouse button. If you change to a floppy drive in this way, make sure that there is a disk in the drive. You will see a warning message if a drive is not available, but this can take some time to deliver and you cannot use DOSSHELL during this time.
- If you often make use of floppy disks along with the hard disk (for backup purposes, for example) you can save time by keeping a disk in the A: drive for as long as you are using DOSSHELL. This avoids any error messages when you are frequently swapping drives.

File Display Options 32

DOSSHELL allows you several ways of using the screen. Place the cursor on the name View in the menu line at the top of the screen and click the mouse button to see the list. The option that is currently in use will not be listed, so that of the following you will see only three:

Single file list

Dual file list

All files

Program/File list

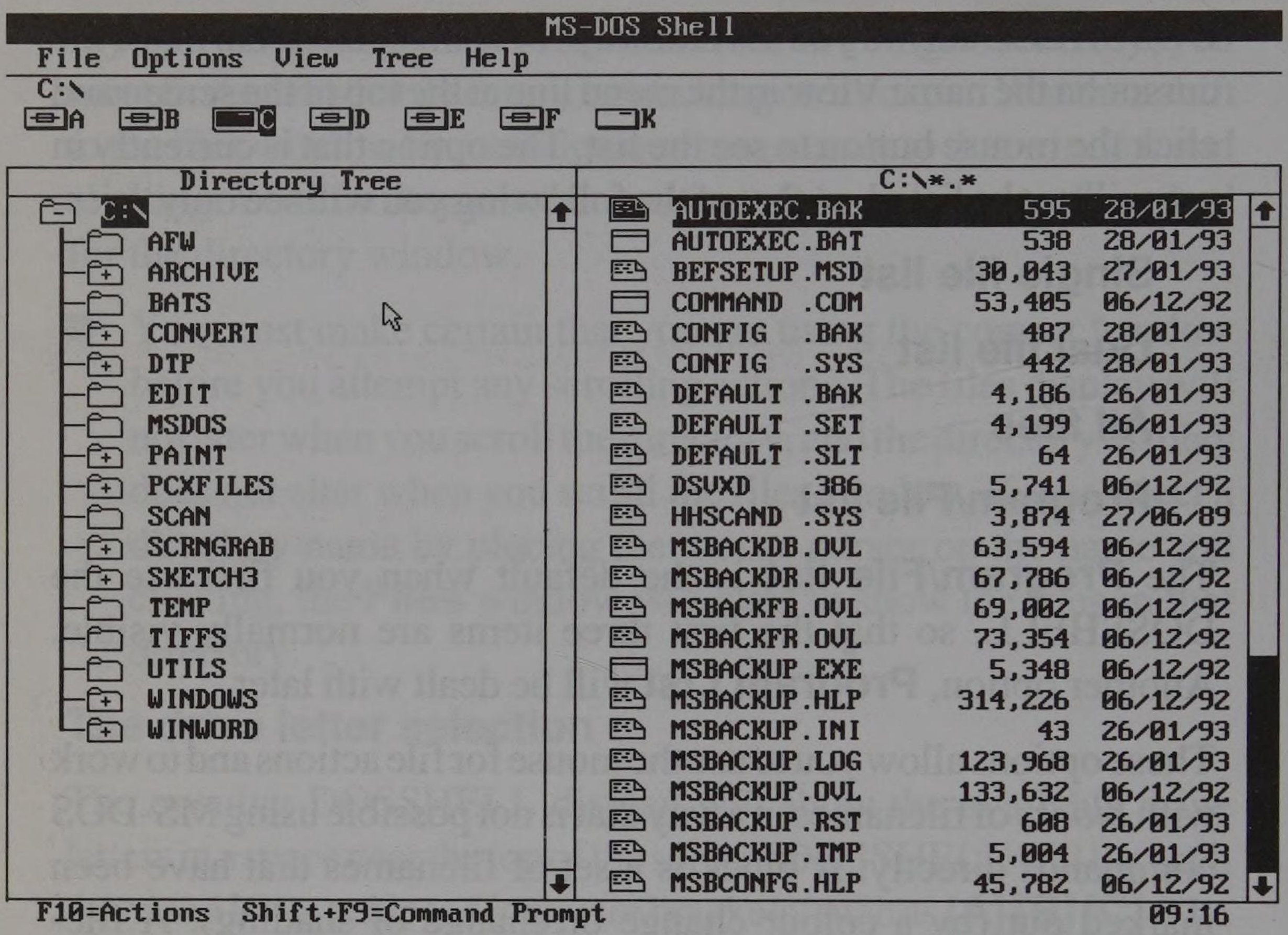
The **Program/File list** is the default when you first use the DOSSHELL, so that the first three items are normally visible. Another option, **Program List** will be dealt with later.

These options allow you to use the mouse for file actions and to work with *blocks* of filenames in a way that is not possible using MS-DOS commands directly. A block is a set of filenames that have been marked out (by a colour change or change of shading). A file-handling command that is used when a marked block exists is applied to **all** of the marked files in the block. You can select these files either from a single directory or from various directories on the disk, see later.

Single file list

The screenshot overleaf shows the form of a single file list. The **Directory Tree** occupies most of the left-hand side of the screen, and the files list starts in the right hand side. You can scroll each list independently by using its scroll bar as described earlier. In this example, you can see the whole of the root directory in one view, a very considerable advantage.

32 File Display Options



A single file list as it appears on screen. This shows directories on the left and the files for the selected directory on the right. The C:\ files are displayed because this is the selected directory.

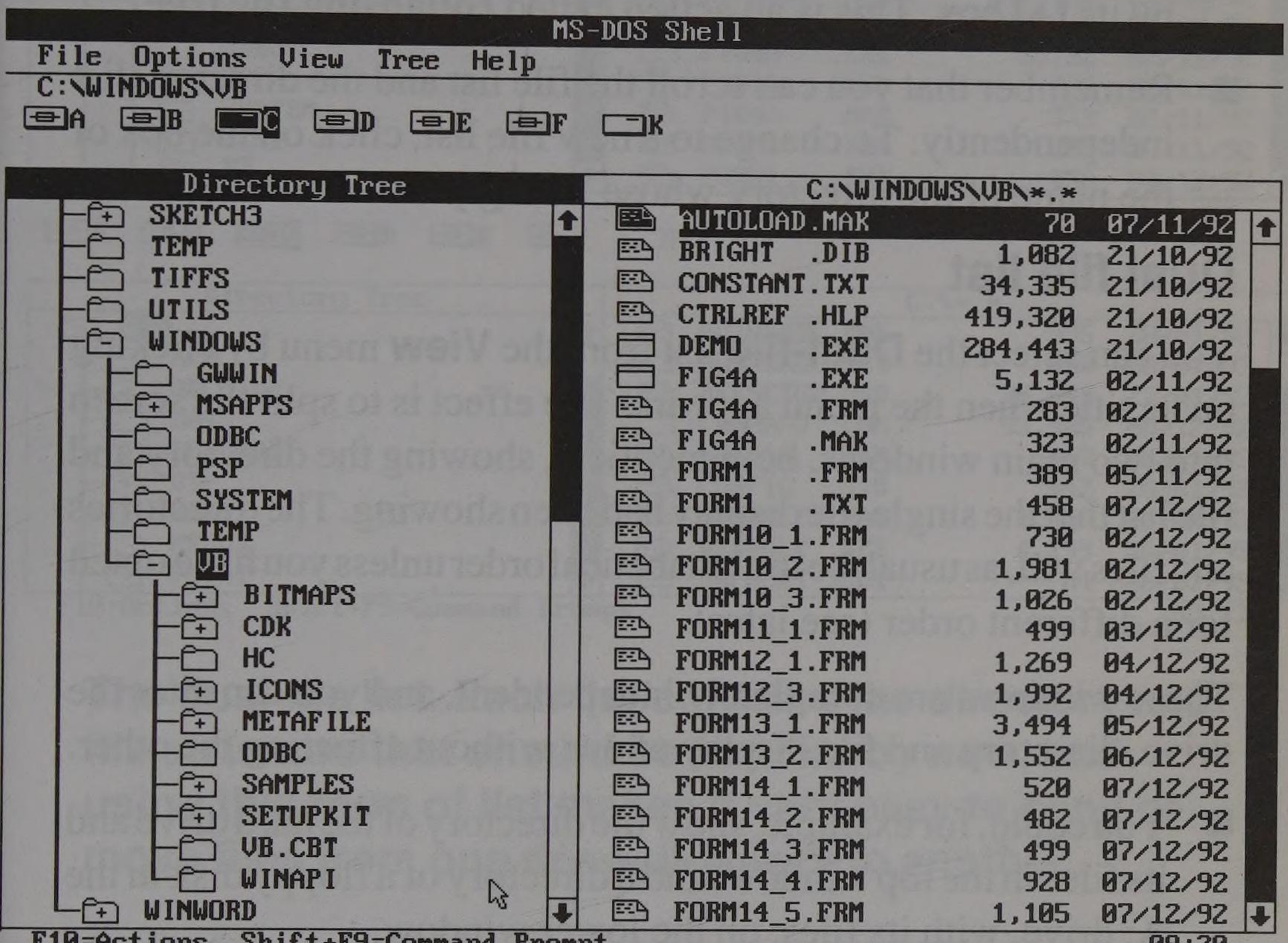
- Even if this is not possible, you normally need very little scrolling of the Directory window to look at all the directories on a hard disk, and it is correspondingly easy to look at all of the files in a directory by selecting that directory.

Each directory will appear beside a status box, a box on the graphics screen or square brackets on the text screen. This box can be empty, or it can contain a **[+]** sign or a **[-]** sign.

1. The **[ ]** box means that the directory contains files, but no sub-directories. Click on the box or on the directory name and you

will see the files of that directory appear in the Files list on the right hand side.

2. The **[+]** box means that the directory contains other sub-directories. Click on the box to see these sub-directories listed under the main directory name.



The presence of sub-directories is indicated by a [+] sign and they can be revealed by clicking on this sign. A [-] sign indicates that the tree has been expanded.

3. Each of these sub-directory names appears next to a box, and some may be marked with a **[+]** to indicate that there are further sub-directories. You can click on the **[+]** box to reveal them, and action called **expanding the tree**.

4. To select files in any of these directories, click on the box or on the name of the directory.

- If you see the **[-]** box this means that the sub-directories are already displayed. You can click on the box to make these names disappear again. This will leave the main directory name marked by its **[+]** box. This is an action called **collapsing the tree**.
- Remember that you can scroll the file list and the directory list independently. To change to a new file list, click on the box or the name of the directory whose listing you want to see.

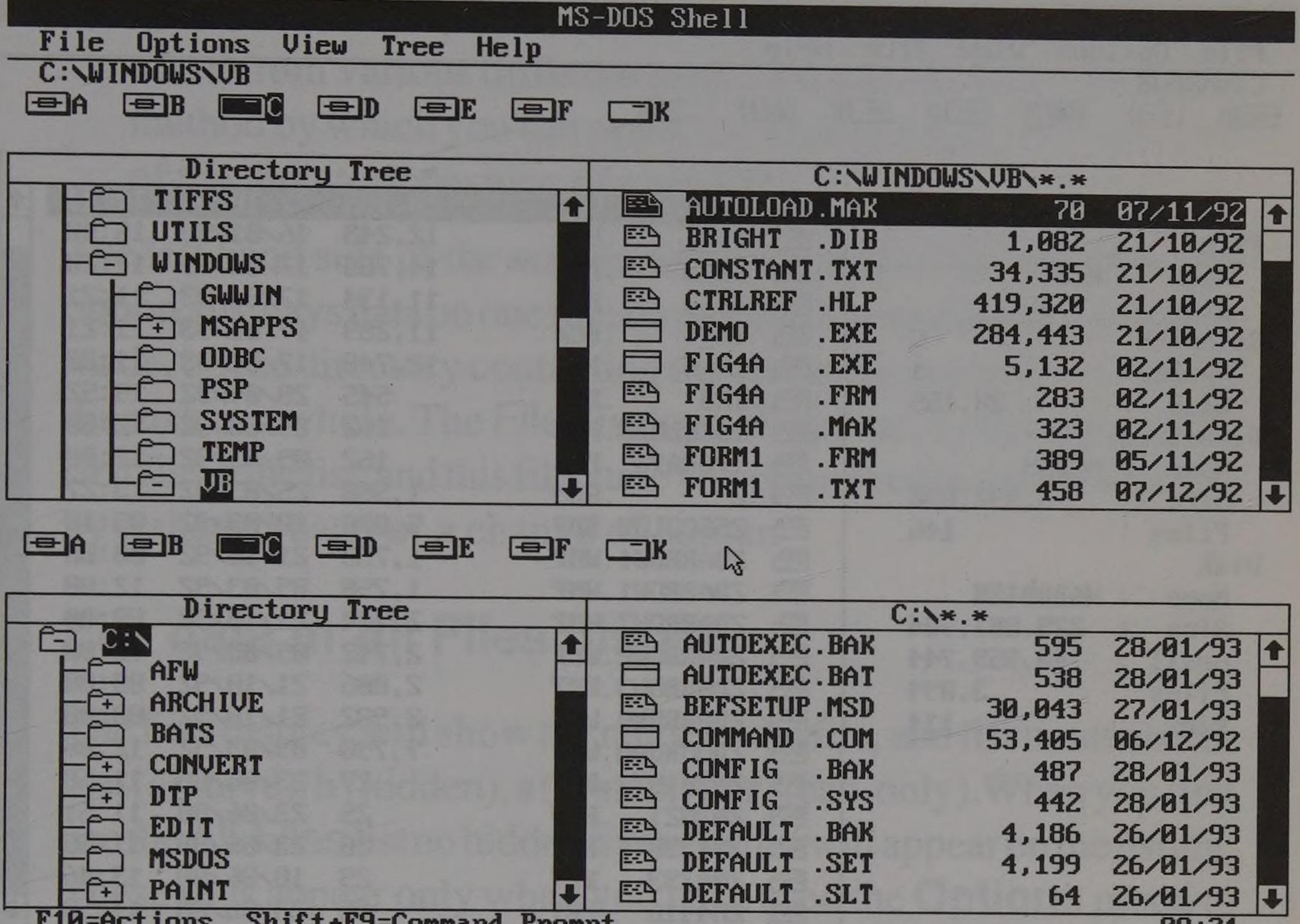
Dual file list

You can select the **Dual-file** list from the **View** menu by clicking on its title when the menu appears. The effect is to split the screen into two main windows, both identical, showing the directory and file list that the single file display had been showing. The directories and files will, as usual, be in alphabetical order unless you have opted for a different order (see later).

These windows are completely independent, and you can alter the drive, directory and file display of one without affecting the other.

- You could, for example, show the directory of the hard drive and its files in the top window, and a directory of a floppy disk in the A: drive, with its files, on the lower window.
- You could equally easily display one hard disk directory and its files in one window and a different directory on the same hard disk in the other window.
- This type of display is particularly useful if you want to copy, move or compare files on different drives or directories, see later.

File Display Options 32

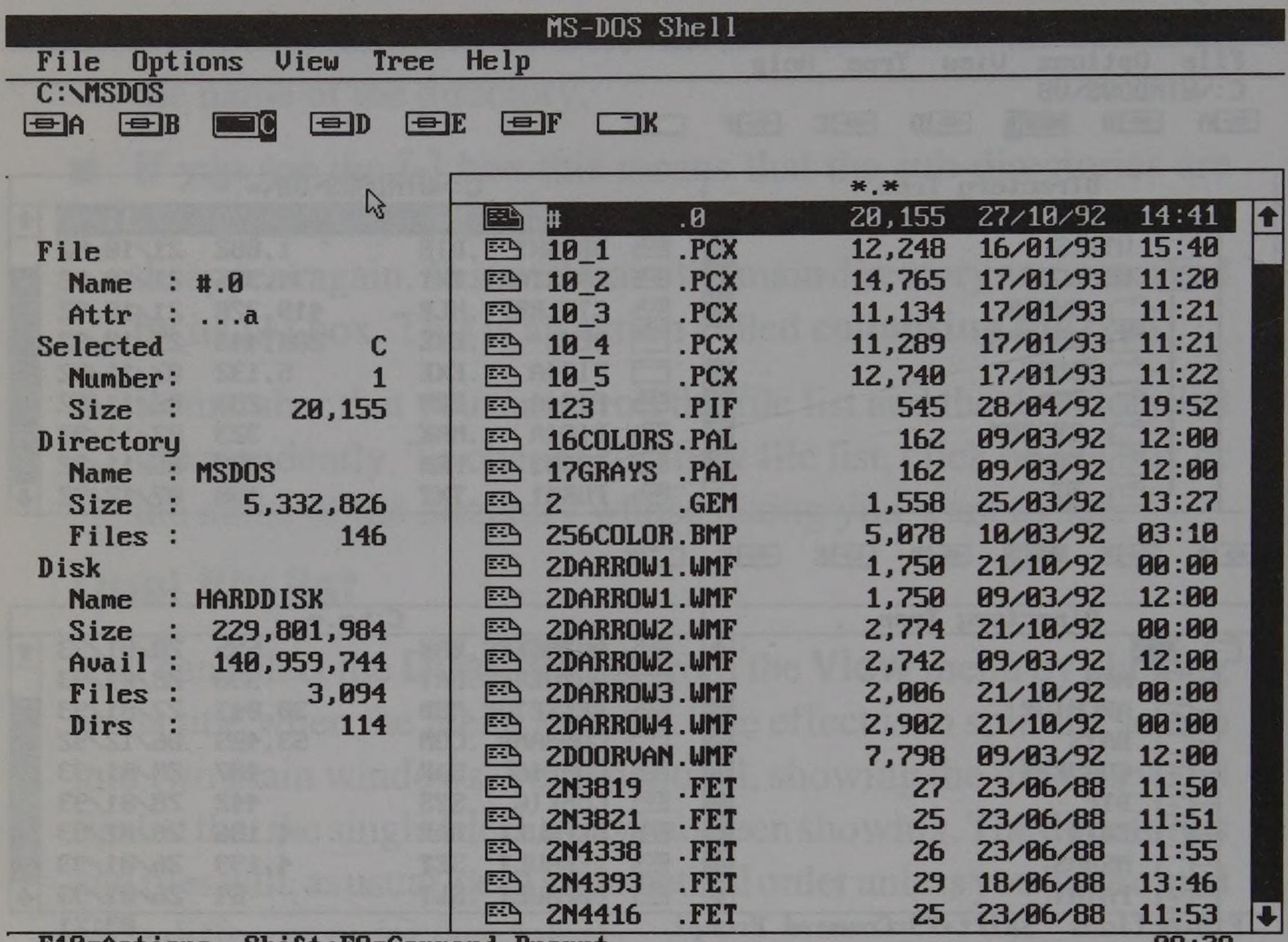


The dual file list. Each list contains the directory and file structure that should be familiar by now, and using this form of list makes it very easy to copy or move files from one drive/directory to another.

All Files list

When you select **All Files** from the **View** menu, you will see a list such as that illustrated overleaf. The files list is on the right hand side, and it consists of **all** the files on the disk. These files will appear arranged in alphabetical order unless you have opted for some other order. If there are files that bear number titles, such as **123.CNF**, the number titles are arranged in order of size ahead of the alphabetical entries.

32 File Display Options



The All Files list of DOSSHELL. This contains all the files of the hard disk (over 3000 in this case) arranged in alphabetical order.

- This listing is of **all** disk files, in order and irrespective of directory. On a hard disk, this listing may be of several thousand files, so that you will need to scroll the files window if you want to see more than a small selection of files.
- Since the files are in order, it is fairly easy to find one that you want. This type of display is particularly useful to find files by name if you do not know which directory to look at. It is also useful to find if files are duplicated, with more than one copy in different directories.

- The All-Files display also allows you to select a block of files taken from various different parts of the hard disk. This is one method by which you can assemble a block of files irrespective of the directory position of each file.

The left hand side of the screen, when you have used the All-Files option, displays data on one file, on any block selection that you have made, on the directory containing the currently marked file, and on the disk as a whole. The File display will appear, by default, for the first file in the list, and this filename will be marked in the file window by a shaded band or a change of colour.

File data in All Files display

The File display will show the name of the file, and list its attributes as **a** (archive), **h** (hidden), **s** (system) or **r** (read-only). When you first use the All-Files list no hidden/system files will appear on the listing. These files appear only when you have used the **Options** menu to show **File Display Options**, and you have selected the option to show hidden files, see later.

The **Selected** portion of the display shows the disk letter, normally C when you are working with the hard disk. Since at least one file will always be selected (the one marked by the shading or colour) there will **always** be an entry for the number of files (default 1) and size of file.

- When you have marked a block of files, the display will show the number of such files and the total size.

The figure for the total size of a set of selected files (a block) is important if you intend to copy the files to a floppy disk. If you attempt to copy more files than a floppy will hold, the action will start, but some files will not be copied.

32 File Display Options

The **Directory** entry shows the name of the directory that contains the currently selected file - this is particularly useful if you find two files of the same name and want to find which directories they use. The **Size** entry shows the total number of bytes used for the whole directory, and the **Files** entry is the number of files in that directory.

The **Disk** part of the display does not change as you select or mark files, because it concerns the disk itself, and will not alter unless you are adding or deleting files. The Disk name is shown, and if you have not given a name (or Volume Label) to your hard disk you ought to do so.

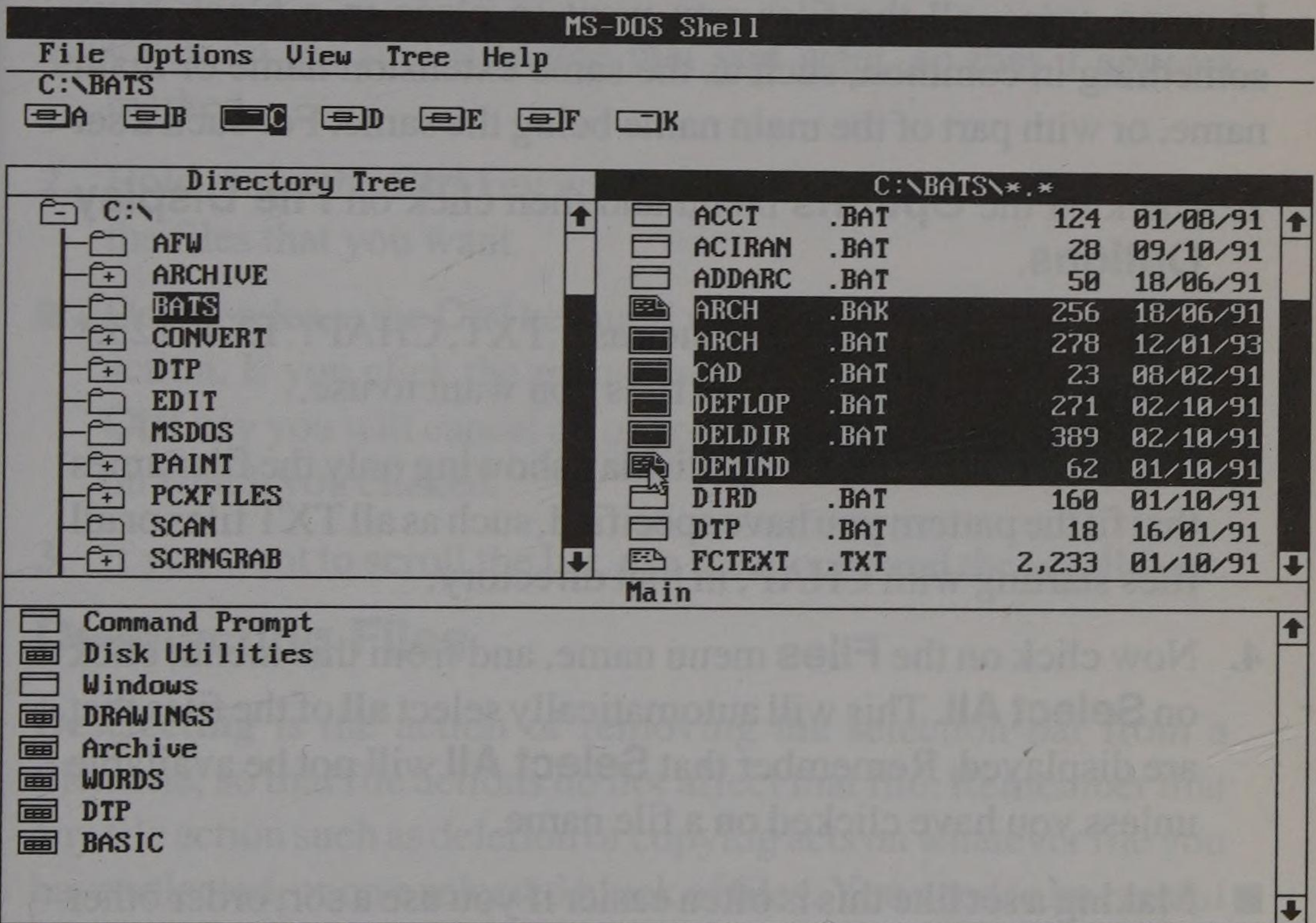
The total size of the hard disk is shown, along with the amount of space that is available. The **Files** entry shows the number of separate files on the disk, all of which, apart from hidden files, will be listed in the **Files** window, and the **Directories** entry shows how many directories have been created (some may be empty).

Selecting Files 33

A **selected** file appears with its name shown with a shaded or coloured bar. You select a single file by placing the mouse cursor over the file name and clicking. There will always be at least one file selected when the mouse cursor is in the files window.

- Do **not double-click** on a data file name unless you have associated the file with a program, see later. If you double-click on a data file that you have not associated with a program this will produce a **Bad command or filename** message. If you **have** associated the file, the associated program will run and will make use of the data file.

The more useful actions of DOSSHELL, however, concern sets of files that you have selected as a block by using the mouse along with other keys.



A complete block marked using the mouse and the SHIFT key together.

33 Selecting Files

You can select a set of consecutive filenames as follows:

1. Place the mouse cursor on the first of the files of the set
2. Click to mark this file name, which will change colour or shading.
3. Hold down the **SHIFT** key, move the mouse cursor to the last name in the set, and click again.
4. This will mark the whole block.

■ If you need to scroll down the list for step 3, use the scroll bars and the mouse, **NOT** the cursor arrowed keys because using these keys will clear the markings. You can scroll in either direction to mark a block, so that you can mark the block starting at the lower item and ending at the upper.

In some cases, all the files you want to place in a block have something in common, such as the same extension name or main name, or with part of the main name being the same. For such a set

1. Click on the **Options** menu and then click on **File Display Options**.
2. Fill in a suitable name line, such as *.TXT, CHAP?.TXT, 123.* or whatever will specify the files you want to use.
3. This will result in the Files display showing only the filenames that fit the pattern you have specified, such as all TXT files or all files starting with CHAP, in that directory.
4. Now click on the **Files** menu name, and from this menu, click on **Select All**. This will automatically select **all** of the files that are displayed. Remember that **Select All** will not be available unless you have clicked on a file name.

■ Making a set like this is often easier if you use a sort order other than by name. You can sort.in order of date, size or other file features, see later.

- Remember that you can switch to the All Files window if you want to select files from different directories. For many purposes, selecting all of the files that conform to a specification like this and then deselecting a few, is faster than picking files from a list to select.

If your selection of files does not conform to a pattern or consist of a set that are arranged in sequence, you will need to select files individually. This is not quite as easy as it sounds, because clicking the mouse button by itself will only ever select the filename on which the mouse cursor is resting; any previous selection will immediately disappear.

Adding to a selection requires you to hold down the **Ctrl** key while you press the mouse button :

1. Click on the first filename that you want, so that it appears marked.
2. Hold down the **Ctrl** key while you click on the other names of the files that you want.

- Do not release the **Ctrl** key until you have finished the selection action. If you click the mouse button again after releasing the **Ctrl** key you will cancel all of your selection except the last file on which you clicked.

3. If you want to scroll the list, use the mouse and the scroll-bars.

Deselecting Files

Deselecting is the action of removing the selection bar from a filename, so that file actions do not affect that file. Remember that any file action such as deletion or copying acts on whatever file you have selected, or on a selected block of files. You need to be careful to ensure that the selection has not changed because of careless clicking.

For a single file, selecting another file name will automatically deselect the other file.

You can **deselect** blocks of files in ways that are very similar to those used to select blocks.

1. All of the selected files in a directory can be deselected by using the **Deselect** option from the **Files** menu. Do this always after you have carried out a file action on a block of files unless you need another block action.
2. When you have selected a block of files you can deselect some files from the end of the block. Place the mouse cursor on the last file you want to de-select and hold the **SHIFT** key down, then click. This will deselect all the files from the cursor position to the end of the block.
3. You can deselect files at random by holding down the **Ctrl** key and clicking on the files that are to be deselected. You can scroll the display by using the scroll bars while this is being done, so that you can deselect over a large range of files.

■ When a file is deselected from a block, the arrowhead will disappear, but the colouring/shading will **not** change until another file is also deselected. This can lead you into thinking that the deselection has not been carried out.

Range of selection

DOSSHELL allows you to select files from different directories that can extend over the whole of the hard disk. There are several methods that you can use.

1. Use the **All Files** list and carry out **File Options** specifications (such as *.TXT or 123.*, for example) to restrict the display to a set of files that you want, taken from any set of the disk directories. You can then use the **Select All** option from the **Files** menu to select these files.

2. Again, using the **All Files** list, you can mark using the **SHIFT** or **Ctrl** keys in conjunction with the mouse cursor as described earlier. You can also use these mouse and key methods on a full list, in which the specifier name is *.* and all files are displayed.

- You can carry out a selection of this kind more quickly than you might expect because the files are arranged in order. Remember that you can determine for yourself the order in which files are displayed; by Name, Extension, Date, Size or Disk order.

3. Use the **Select Across Directories** option in the **Options** menu. This is an on/off switch item, and is marked with a dot when it has been switched on (click again on the name to switch it off and make the dot disappear).

- When you have switched on **Select Across Directories**, you can make selections, as described earlier, using the mouse along with the **SHIFT** and **Ctrl** keys as you wish. You can, for example, click on a single file name in one directory, then change directories and use the **SHIFT** key along with the mouse to mark out a consecutive set of files. You can then change directory again and pick out some scattered filenames to mark, using the **Ctrl** key along with the mouse button.

All of the names that you have marked in this way will remain marked unless you return to a directory and mark some other name(s). When you have marked a set of files in different directories like this, all of the usual file actions can be applied, as listed later.

- You have to use **Select Across Directories** with some care. If you leave it switched on it is only too easy to move from one directory to another leaving a trail of selected files behind you. If at some stage you decide to delete a file you are liable to delete the whole set of files you have marked. For safety, use **Select Across Directories** only when you need to, and always

switch it off when you have finished using it. In addition, keep the **Confirmation** options, see later, switched on. This ensures that you will be notified of each file that is to be affected, and you will know if a file appears from a different directory.

View File Contents

You can use **View file contents** (File menu) only when you have selected a single file. This option is not visible in the File menu if you have selected a block of files, or if the File list is not in use. The alternative method of pressing the **F9** key will work only if a file is selected.

View File Contents is an excellent way of displaying the words in batch or other text files, whether these have been generated by a text editor or from a word-processor.

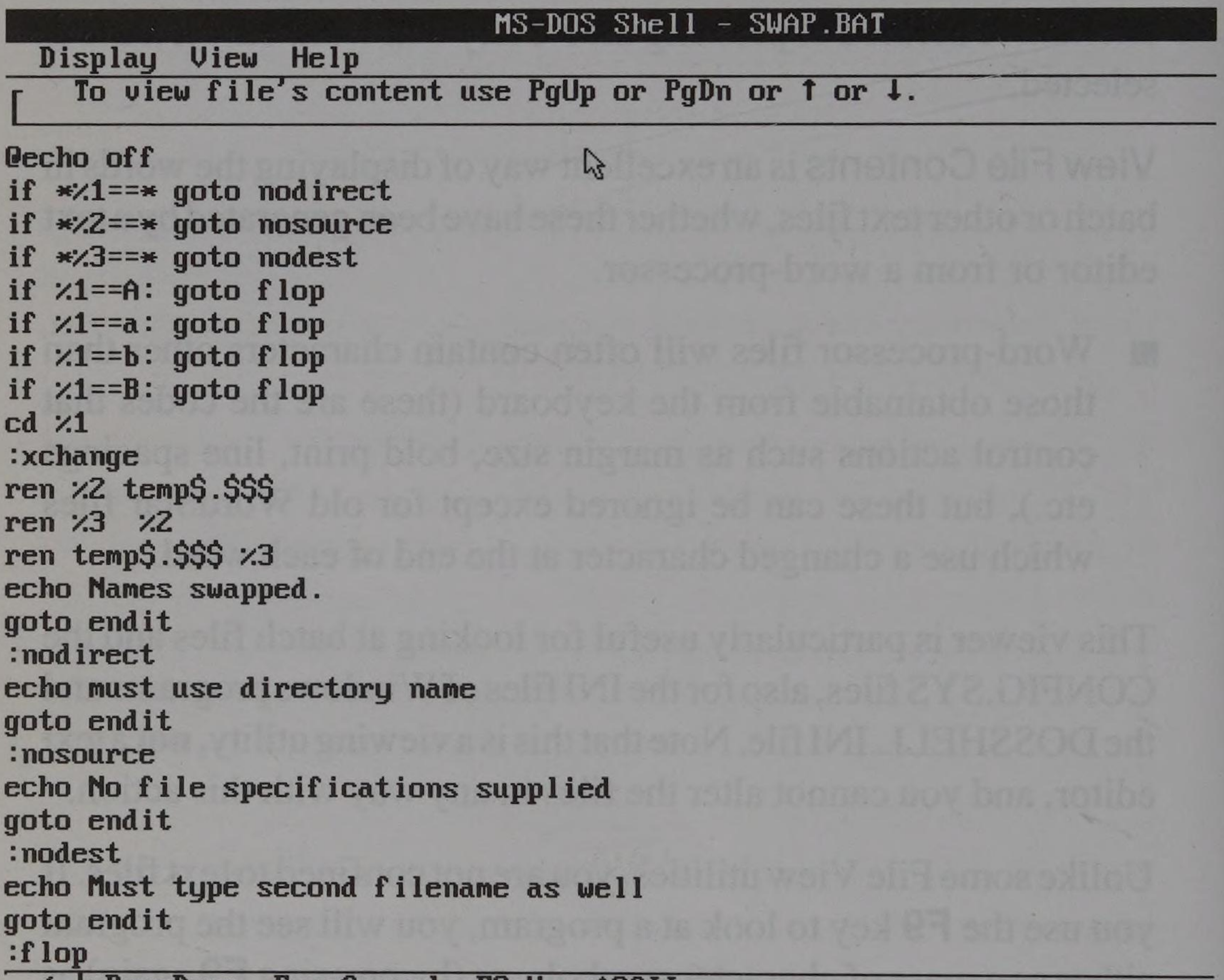
- Word-processor files will often contain characters other than those obtainable from the keyboard (these are the codes that control actions such as margin size, bold print, line spacings etc.), but these can be ignored except for old WordStar files which use a changed character at the end of each word.

This viewer is particularly useful for looking at batch files and the CONFIG.SYS files, also for the INI files of Windows programs and the DOSSHELL.INI file. Note that this is a viewing utility, **not** a text editor, and you cannot alter the files in any way with this action.

Unlike some File View utilities, you are not confined to text files. If you use the **F9** key to look at a program, you will see the program either as a mass of character symbols, or (by pressing **F9** again) as a set of numbers in hexadecimal scale.

The display switches over each time you press the **F9** key, so that you cannot be certain in which mode it will start when you use View File Contents on a new file. If you see mainly a set of hexadecimal numbers such as F8, 3A, etc., when you want to view a text file, press **F9** to restore the full character display.

The View File Contents utility will scroll in the usual way with the cursor arrowed keys or the **Page Up, Page Down** keys. You can also use the **ENTER** key as a **Page Down** key. You can also click with the mouse on the words **PgUp** and **PgDn** or the arrows at the top of the screen.



A batch file displayed using the F9 key. Scrolling can be done by clicking on the PgUp, PgDn or arrow signs at the top of the screen or by pressing these keys.

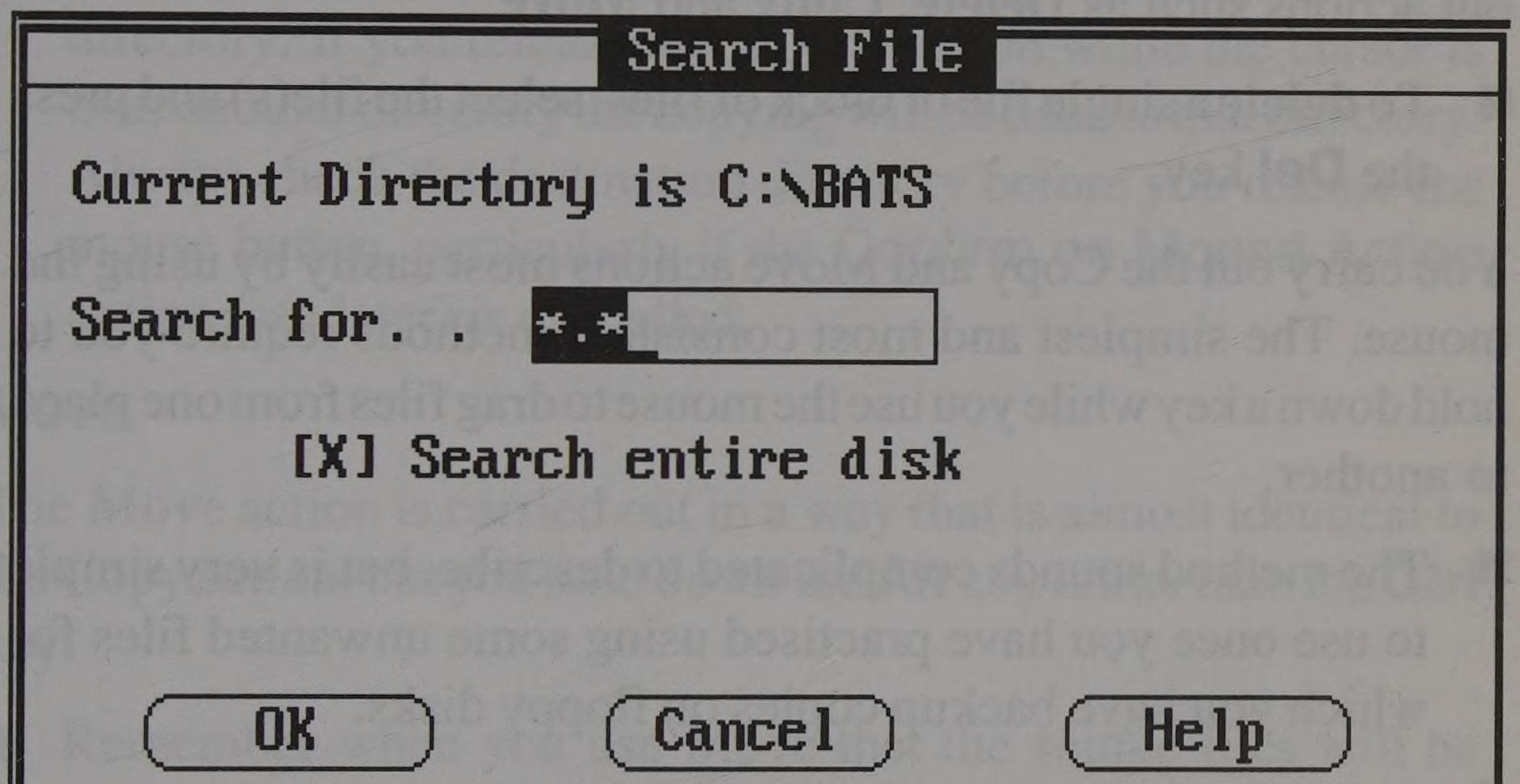
You can leave the File View utility by pressing the **Esc** key, or by using the mouse to select the **Restore View** from the little **View** menu. The use of the **Esc** key is quicker and simpler.

File Search

DOSSHELL contains, in its Files menu, a **Search** action which you can use to find a file, either in a selected directory or using a search of the entire disk. You have to specify what you are searching for, and supply a valid filename which can include a wildcard.

- You must place the cursor in the Files window to allow this option to appear in the **Files** menu - it is not available from other windows.

1. Click on **Search** to bring up the **Search Form**.



The Search form which allows you to find your specified file in a selected directory.

2. Type the name of the file and indicate whether the search is to be over the whole disk (the default) or in the selected directory only.

- You can search for a filename, but not for a phrase or word within a file, so that the name that you supply must take the form of a valid file name (8 characters maximum in the main name and up to three in the extension).

3. When the file is found, its name will be printed on a report form.

- If the name contains wildcard characters, several names will be printed which match the pattern you have typed.

Mouse methods

You can select a single file or a set of files, see earlier, and then carry out actions such as **Delete, Copy** and **Move**.

- To **delete** a single file or block of files, select the file(s) and press the **Del** key.

You carry out the Copy and Move actions most easily by using the mouse. The simplest and most consistent methods require you to hold down a key while you use the mouse to drag files from one place to another.

- The method sounds complicated to describe, but is very simple to use once you have practised using some unwanted files for which you have backup copies on floppy disks.

Copy

A **Copy** action is done, typically, like this:

1. You switch to a view that allows you to see both source and destination drive and directory, using the Dual Files list.

2. With the cursor in the **Source** directory (which can be either window - whichever directory you are copying from), select the single file or block of files which must be copied.

3. Now hold down the **Ctrl** key, place the cursor on any marked file in the source set, and hold down the mouse button.

4. Move the mouse until the cursor is over the **Destination** directory. This must be on the left-hand side of the destination display, over the directory name and **not** over any filename on the right-hand side.

5. Release the mouse button, then the **Ctrl** key, in that order. The files will then be copied and you will see the directory contents alter.

■ You must be careful that you select the correct destination directory. If you release the mouse button while the cursor is over another directory the copying will be done to that directory. Always check the destination directory before you release the mouse button, particularly if the Confirm on Mouse Action option (see later) is cancelled.

Move

The **Move** action is carried out in a way that is almost identical to the Copy action, but you hold down the **Alt** key rather than the **Ctrl** key.

■ Remember when you use Move that the source files will be deleted after copying. This action is automatic and you cannot interrupt it. There should be no risk of losing files in the event of a power-failure, however, because the deletion is not done until the copy action is complete.

Both Copy and Move actions will require you to confirm for each file in a selected block unless you have turned off the Confirmation messages in the Options menu. While you are learning to use these commands, it is better to keep the confirmation messages in place, but you can cancel the confirmation when you are more sure of the actions. This will make the actions very much faster.

34 File Actions

- If you drag files from one directory of the hard disk to another, with **no** keys held down, this carries out a **Move** action. Dragging files from one disk to another (between a hard disk and a floppy usually) carries out a **Copy** action only. You can make this into a **Move** by holding down the **Alt** key as you drag the files.

Part of the Files menu contains a set of file actions that you can carry out either on individual files or on blocks of files selected by any of the methods that were described earlier. These actions offer you an alternative to the use of the mouse, and the list is:

Move

Copy

Delete

Rename

Change Attributes

Of these actions, **Rename** and **Change Attributes** have no mouse equivalents, so that we will look in detail only at these actions.

The

action is oddly restricted in a way that does not apply when you rename a set of files by using MS-DOS commands directly.

1. For a single file, select the file so that it appears marked.
2. Now click on **Rename** from the Files menu.
3. This brings up a **Rename** box into which you type the new name and clicking on OK will cause the rename action to work.

- To rename a whole set of files, such as CH1.TXT, CH2.TXT, CH3.TXT, etc. to CH1.DOC, CH2.DOC, CH3.DOC and so on, you must select all the files that you want to change, and start the **Rename** action as before, but you will prompted to type the new name for each file individually.

This way of renaming a set of files is tedious, and the old-style MS-DOS command is much faster, see Section 28.

You should use the DOSSHELL **Rename** for single files, and for groups of files for which each new name differs completely from the

old name. Use the MS-DOS **REN** command when the old and new filenames differ only in a straightforward way such as DOC for TXT, ART1 for CHAP1 and so on.

You can also change the attributes of files very easily from DOSSHELL, more easily than from MS-DOS commands. You should use this very sparingly, and mainly for making files read-only.

Associating data files

A data file is not a program, so that double-clicking on a data file cannot normally make a program run. This applies whether it is a text file or one that has been produced by a spreadsheet, a database or graphics programs such as CAD or Paint programs. Such files are useful only when one is loaded into a running program of the correct type.

DOSSHELL provides for you to **associate** any data file with a program. When you have done this, double-clicking on the data file will start the program and read in the data. The action is just as if you had double-clicked on the program name and then opted to read the data file from the program.

- Association works only if you use consistent file extensions. It would be pointless if your data files did not use extensions, or if you used the same extension for all data files.

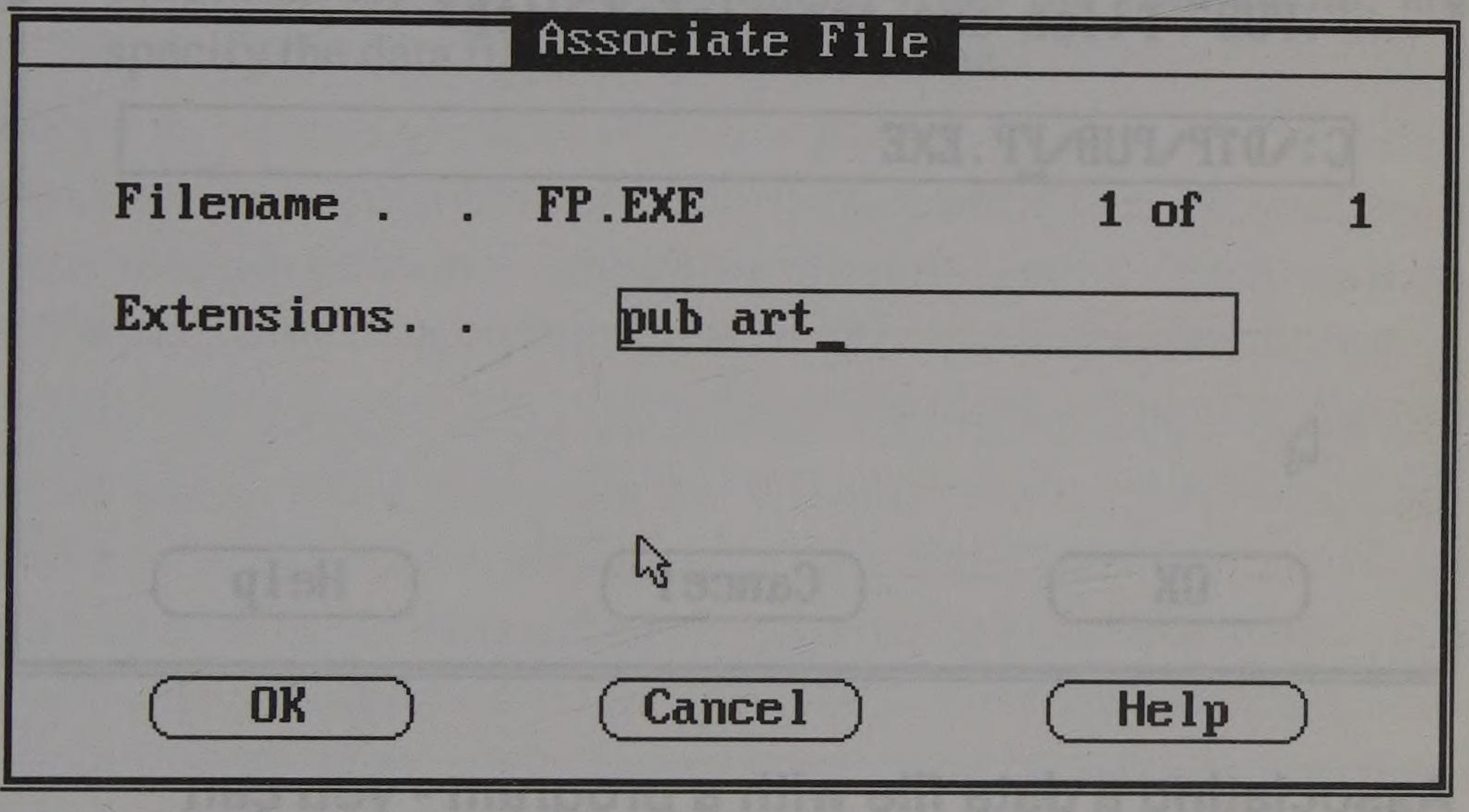
Not all data files can be Associated. What determines if association is possible is the program that produced the file. For example, suppose you have a program called TEXTMAKE. If you can, using MS-DOS directly, type the line:

TEXTMAKE myfile

to make this program create a file called *myfile*, then you can use association on the files that this program creates. If this system cannot be used, no association is possible.

- Associating data files provides you with a timesaving way of working with your data files, since you do not need to locate the program file in order to make use of the data file once the you have made the association. You can specify association either for a program file **or** a data file. For a **program** file:

1. Select a program file in the usual way (remember to click once only).
2. Click on the **File** menu name, and then on **Associate**.
3. A form appears. You are asked to type the extension letters to look for. Data files with these extension letters will be associated with the program file. Click on **OK** to create the association.

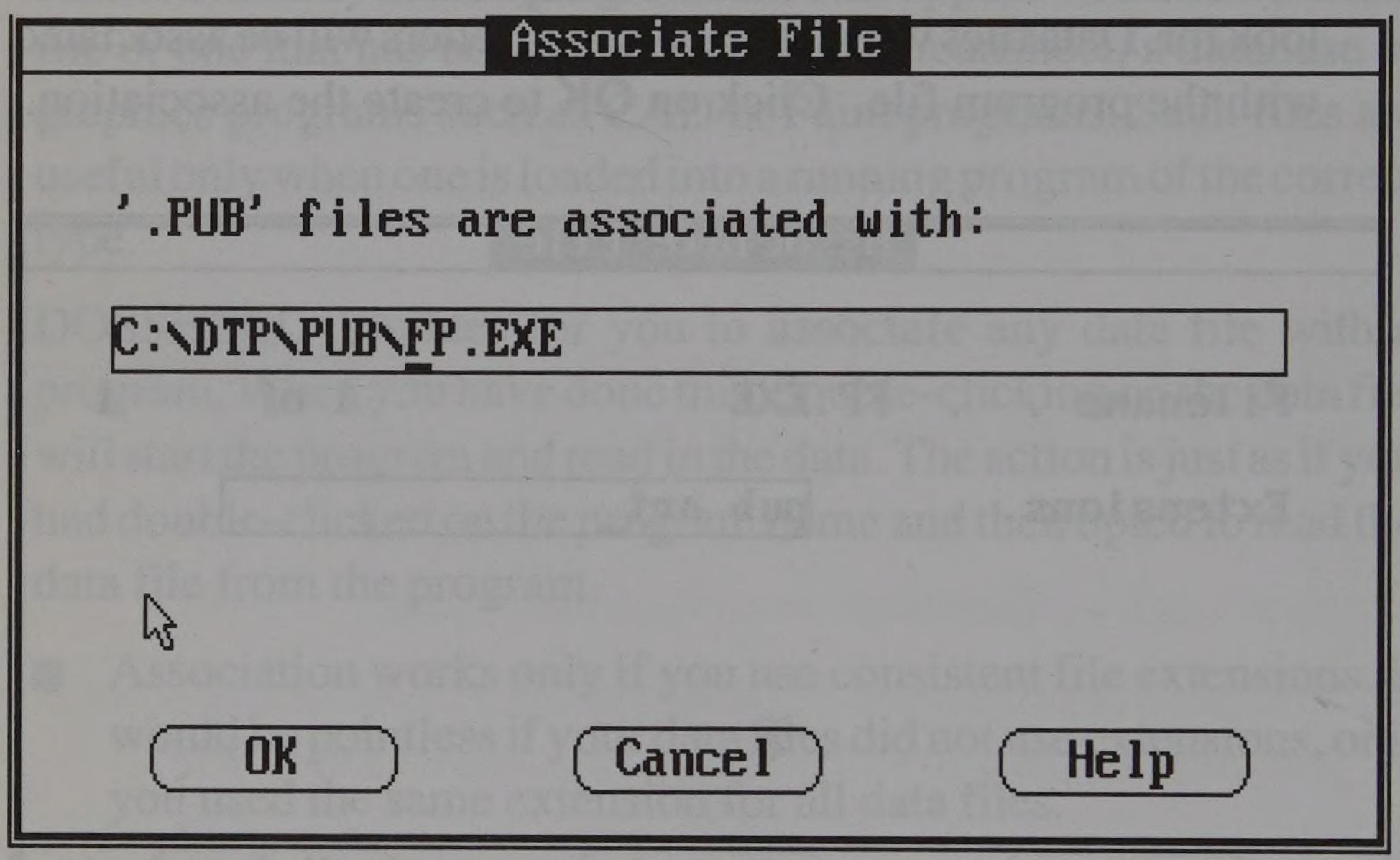


The Associate File option being used on a program name, requiring you to type in an extension that the program uses for its data files. You can use more than one extension.

4. You can type several sets of extension letters separated by a space. In future when you double-click on such data files the program will start and load the data file you have selected.
5. In this respect, a batch file is counted as a program file, but you should **not** associate data files with a batch file.

To associate with a **data** file:

1. Select a data file of the kind you want to associate.
2. A form will appear. You will be asked to type a program name.



Associating a data file with a program - you can associate with only one program.

- You must provide the **full path** to the program file so that it can be located from any directory of the hard disk.

3. You create the association when you click on OK.

- A **batch** file is treated as a program file, not as a text file, because double-clicking on a batch file will run the batch file. You should not attempt, therefore, to associate batch files with any text-editor program to allow you to edit them more easily.
- You cannot successfully associate one type of data file with more than one program. Be careful also of creating conflicts by using one association in a program file and a different one in a data file.
- You can remove an association by selecting the file, using the **Associate** option, and deleting the name(s) from the list.
- If you want to use a data file along with another un-associated program, use the **Run** command from the **File** menu, and specify the data file in the command line.

36 Program File Running Options

You will normally run a program by double-clicking on its name (see later for program Groups). This, however, does not allow you to enter any information. For example, using MS-DOS directly you might start a program called PIXIT by using the line:

PIXIT /M /DXT

in which the **/M** and **/DXT** are parameters. In this imaginary example, using **/M** might indicate that a monochrome display was required, and using **/DXT** might mean that the program should work with data file using the DXT extension.

Double-clicking on a program name does not allow you to enter these parameters, only to run the program. If you always want to use the same parameters, placing the program in a Group, see later, allows for the same parameters to be used each time you double-click on the program.

- If the program permits you to use a filename as a parameter, you may be able to *associate* file extensions with the program, and start the program by double-clicking on the name of the **data** file.

The **Files** menu of DOSSHELL contains the **Open** and **Run** options, which you will normally use to a much lesser extent than the other commands. **Open** is the equivalent of double-clicking on a program name; it will run that program with no provision for adding parameters. **Run** is more useful - it provides a small panel in which you can type the program name and any parameters that are associated with it.

For any programs that you run frequently it is much more useful to set up a Group, see later, so that you can run, by double-clicking, the program and any associated batch files. **Run** is useful for programs that you might want to run every now and again.

The Group System 37

DOSSHELL can deal with programs in a way that will be unfamiliar to many new users of MS-DOS 6.0. The idea is that programs which belong together can be placed into **Groups**, displayed separately in DOSSHELL, and you can run each program in a Group by double-clicking on its name. What is more important, you can control the way that a program runs by setting up some items, rather in the way that you set up a batch file (see Sections 48 - 51), when you create the Group, or at any later time. Programs that you have placed into Groups can be very easily task-swapped (see later) using a simple key combination for each program.

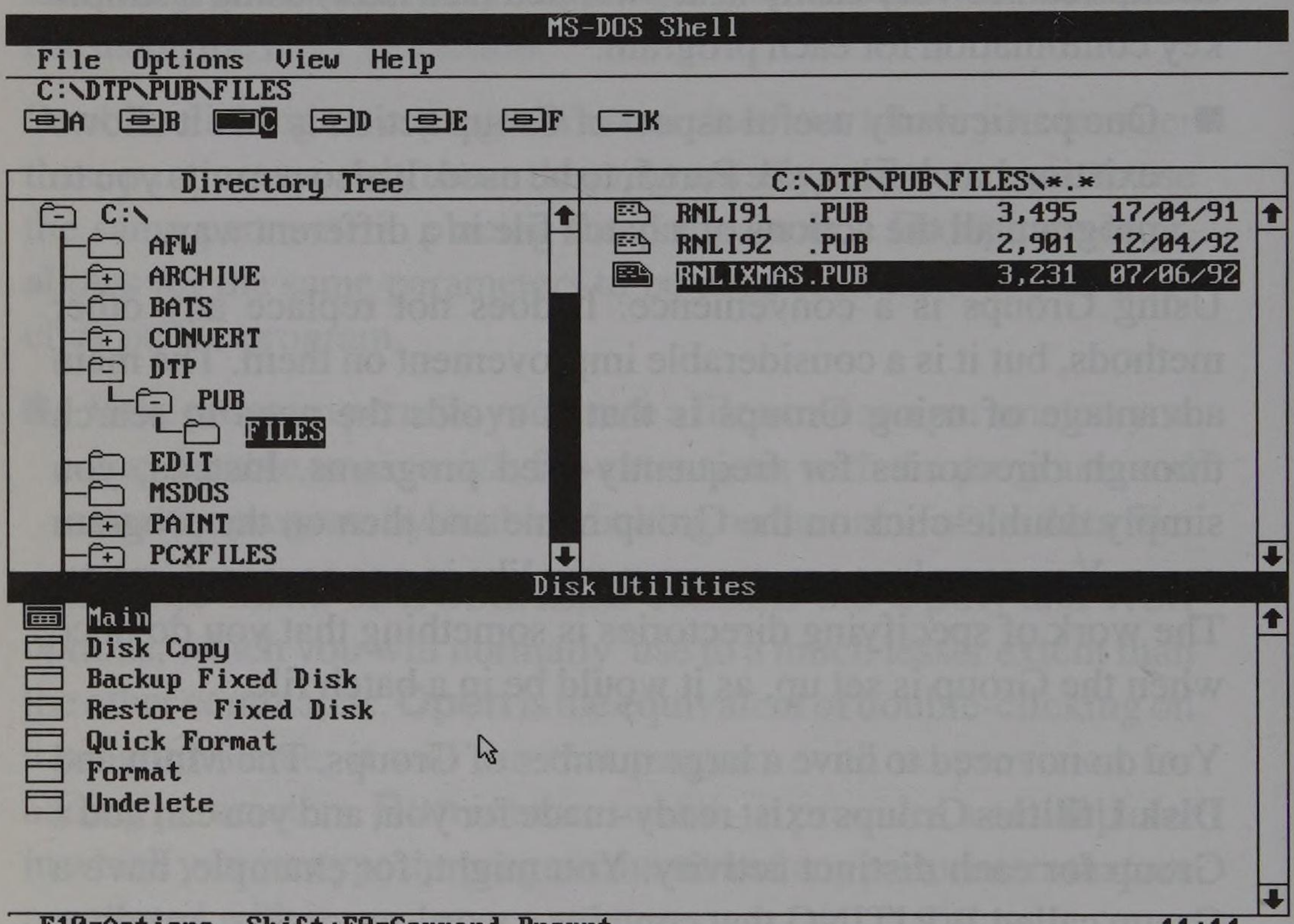
- One particularly useful aspect of Group action is that it allows existing batch files, see Part 5, to be used. It also permits you to program all the actions of a batch file in a different way.

Using Groups is a convenience. It does not replace any other methods, but it is a considerable improvement on them. The main advantage of using Groups is that it avoids the need to search through directories for frequently-used programs. Instead, you simply double-click on the Group name and then on the program name. You can place any program you like in one or more Groups. The work of specifying directories is something that you do once when the Group is set up, as it would be in a batch file.

You do not need to have a large number of Groups. The **Main** and **Disk Utilities** Groups exist ready-made for you, and you can add a Group for each distinct activity. You might, for example, have a Group called WRITING that contains a word-processor, an editor and some file utilities, and another called ACCOUNTS that contains a spreadsheet, an accounts program and some file utilities. This might be as much as you need. It is unlikely that you would need a very large number of Groups, certainly not as many Groups as you have directories on a hard disk.

Existing Groups

When you start DOSSHELL for the first time there are some ready-made Groups and, of these, the **Disk Utilities** Group is the obvious one to select as an example, since everyone will need to use one of the programs in this Group at some time. Double-clicking on this name will bring up the list of programs covered by this Group, and also the name **Main** that allows you to get back (by double-clicking on **Main**) to the original list.

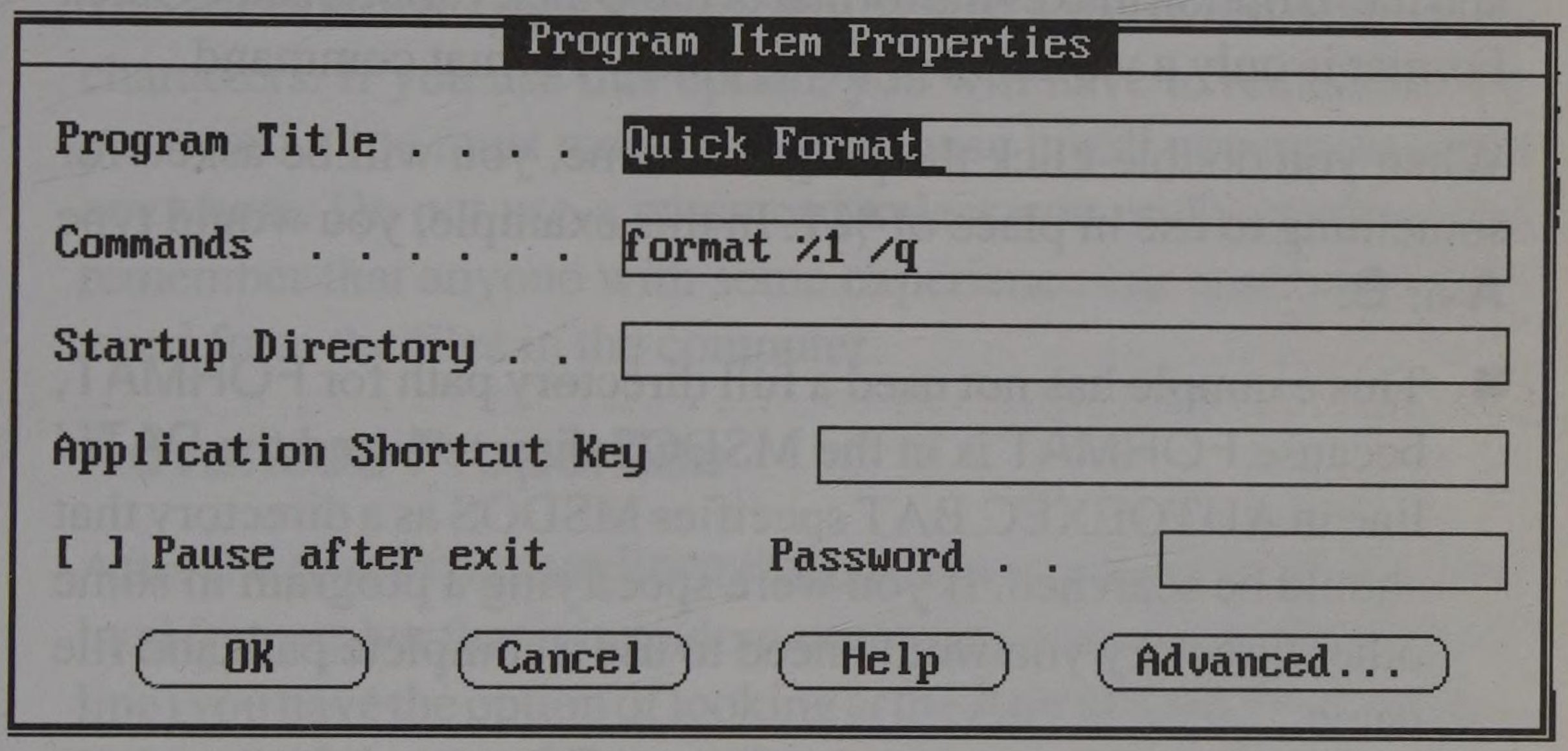


Programs in the Disk Utilities Group - these are set up ready for you and are used here as examples of Group structure.

Each line in the list of Disk Utilities can be used to summon up a program. This in itself is straightforward, and in some cases is no more than could be done by double-clicking on the program name

in the Files list. The difference is that you do not have to search through directories to find the program, and you can specify how you want to make use of the program in a way that is not possible when you simply double-click on the name from the Files list. To see how this is done:

1. Select the **Quick Format** program by a single-click (NOT a double-click) on its name.
2. You will see the name change colour and an arrow will appear to the left of the name.
3. Now select the **File** menu, and click on **Properties**. This will bring up the **Program Item Properties** window



The Program Items Properties window. This allows you to alter the way a program is used from its Group. The example shows the PIP for the Quickformat command, with FORMAT %1 /q used as the command.

4. This is the format that allows you to select options, and in this case, some of the form has been used (as much as is needed in this case). You can edit each line while the form is in use, but in this example you should not alter the information.

The **Program Title** line is for information only - it decides what will appear in the Group as a title. This need not be the same as the filename for the program, and it can take up more space than a filename, allowing you to use a longer description, up to 36 characters if need be.

The **Commands** line contains the commands as you would type them if you were using Run. In this case, the command is **FORMAT %1 /q**. In this example, the **%1** is a placeholder for a drive letter, and the **/q** option makes the format of the Quick variety, since Quick Format is only a variation of the ordinary Format command.

When you double-click the program name, you will be asked for something to use in place of **%1**. In this example, you would type **A** or **B**.

- This example has not used a full directory path for FORMAT, because FORMAT is in the MSDOS directory and the PATH line in AUTOEXEC.BAT specifies MSDOS as a directory that should be searched. If you were specifying a program in some other directory you would need to use a complete path and file name.

The **Startup Directory** line is not used in this example. It allows you to specify a directory, usually one that holds data files, from which the program will be started. If you were using a word-processor, for example, you might want the Startup Directory to be the one that contained the text files. This allows the program instant access to those files when it starts running. Not all programs can use this facility.

The **Application Shortcut Key** is also left blank in this example. Specifying a shortcut key allows program switching (see later, this Chapter) to be done very rapidly by pressing key combinations such as Alt-A, Alt-B and so on. A Shortcut key can be specified **only** for a program in a Group.

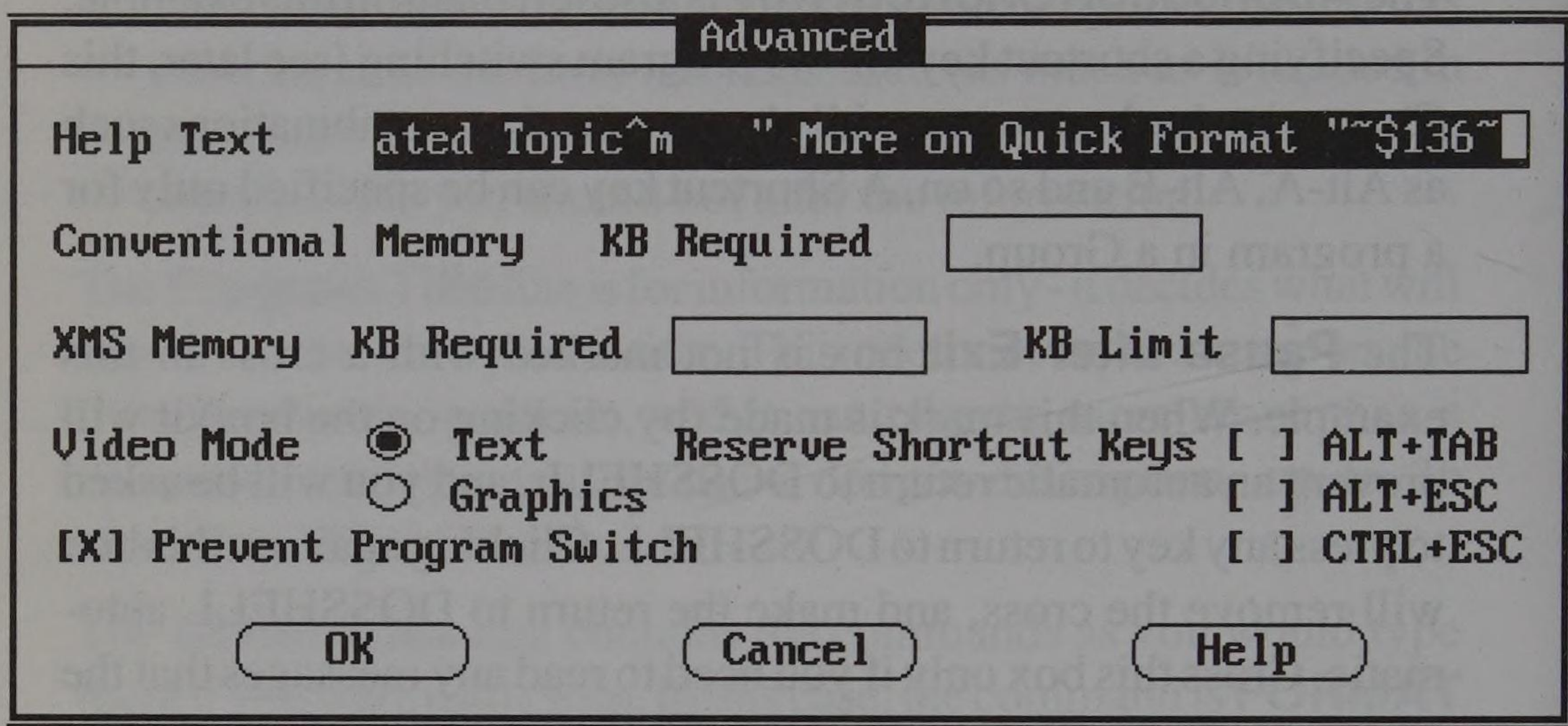
The **Pause after Exit** box is not marked with a cross in this example. When this mark is made (by clicking on the box) it will prevent an automatic return to DOSSHELL, and you will be asked to press any key to return to DOSSHELL. Clicking again on this box will remove the cross, and make the return to DOSSHELL automatic. Cross this box only if you need to read any messages that the program ends with.

The **Password** space allows you to type a password of up to 14 characters. If you use this option, you will have to remember the password - you must make a note because it will not appear again anywhere. Do not use a password unless you really need to, and remember that anyone with some experience can read your password from the files in the computer.

Advanced Properties

After the first Program Item Properties window has been examined (remember that it is seldom necessary or useful to fill in every line) you have the option of looking at the **Advanced Properties**. Using the Advanced Properties is not essential, and some of the options are not really advanced. Since we are looking at an existing set it helps to see how this window is used.

Select **Advanced** by clicking on the name. The Window now looks as illustrated overleaf.



The Program Items Advanced Properties window with its Help text and specifying in this example the use of the Text mode screen.

Of the items shown, the most obviously useful is the **Help Text** line. This is what you will see if you take the **Help** option when you see the form that appears when you start a program from a Group. In this example, the Help Text has been provided. This text takes up more space that is available in the line, but you can scroll it sideways with the left and right arrowed keys, and with the Home (start of text) and End (end of text) keys.

- When you place a program into a Group for yourself, you can use this line to write your own Help messages, which can be as brief or as detailed as you like. It is always useful to type some Help text, because you tend to forget precisely what you must do when running a program, and this text, together with the prompts that you can add, assists you greatly.

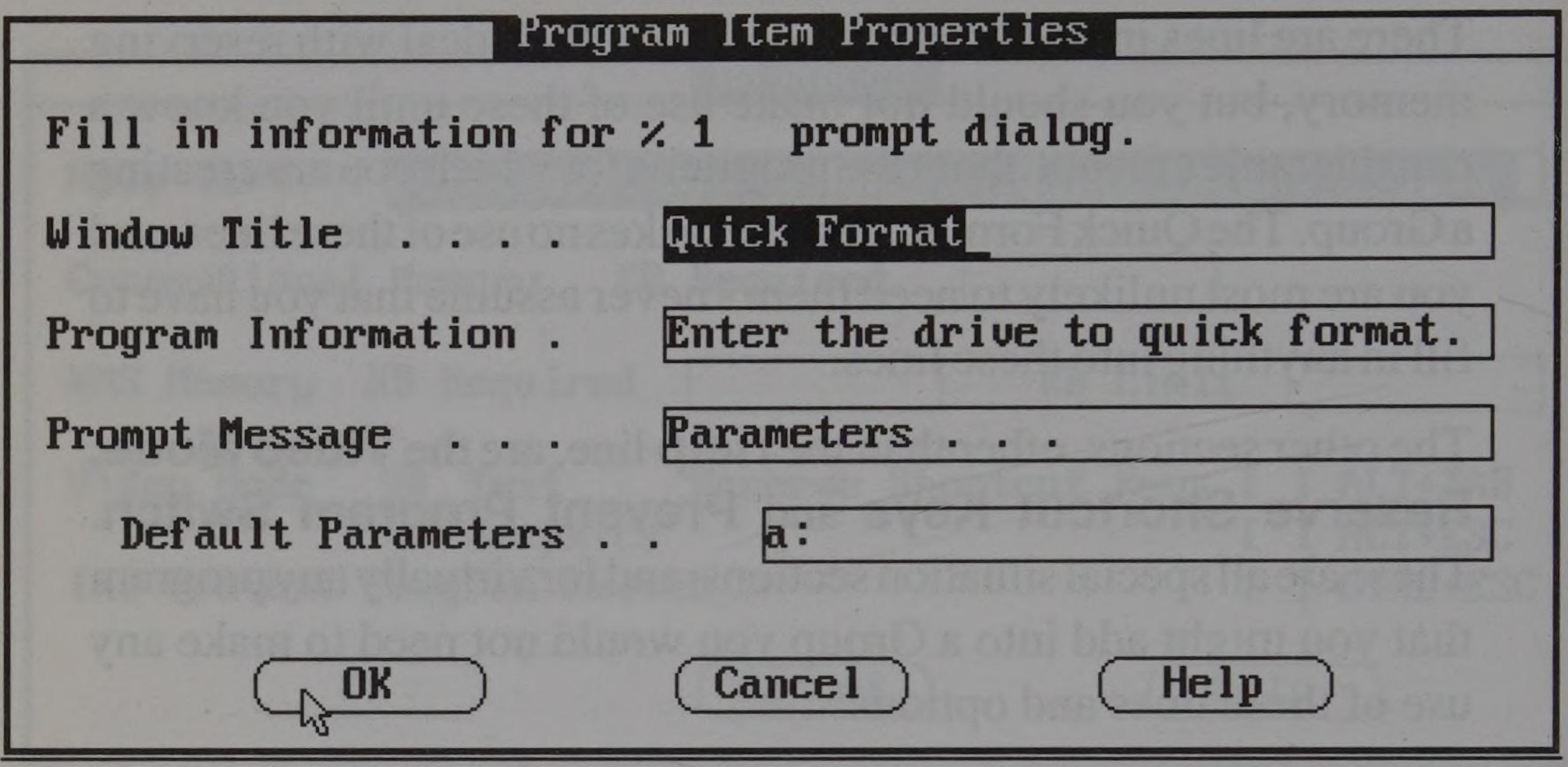
There are lines in the Advanced options which deal with reserving memory, but you should **not** make use of these until you know a considerable amount about the programs for which you are creating a Group. The Quick Format example makes no use of these lines, and you are most unlikely to need them - never assume that you have to fill in anything into these lines.

The other sections, other than the **Help** line, are the **Video Mode, Reserve Shortcut Keys** and **Prevent Program Switch**. These are all special situation sections, and for virtually any program that you might add into a Group you would not need to make any use of these lines and options.

When you click on the **OK** (or **Cancel**) box of the Advanced form, you are returned to the ordinary Program Item Properties form, and at this point you would normally have filled in everything that was needed if you were adding an item.

There will, however, be some further forms to fill in if you have opted for any parameter items in the Commands line, such as **%1, %2** and so on. These **%1, %2** represent items that you normally have to specify in order to run a program (such as filenames, directory paths and so on), and a separate form will appear for each of these. The syntax is exactly as it is for batch files.

In the Quick Format example, there is only one such item, so that when you click on the **OK** box of the main Program Item Properties form, the subsidiary form, appears. This uses four lines, and you would normally make use of all four as has been done in this next example.



The subsidiary form for dealing with a parameter. There will be one box of this type for each parameter, and you can enter reminders in the form of text in the three lines illustrated.

- If you need to edit a line, be careful about your use of keys. When the line appears in inverse colours, pressing the backspace (delete) will delete the whole line. Pressing a left or right cursor arrow key will restore normal colours and allows you to delete or replace individual characters.

The **Window Title** is a reminder of what the program is called - you would normally use the same name (which will appear as a default) as you had used in the main Program Item Properties form, but if you want to use some other title you are free to do so. The **Program Information** line is the more useful one - you can type into this what information you need. In this example of Quick Format, the drive name is required, and this is stated in the Information line.

The **Prompt Message** should be brief - it will appear on a line just ahead of the space where you will type the item, in this example the drive letter. In this example, the word *Parameters* has been used, but you would probably want to be more specific. For any Format, remember, you can type a large number of parameter letters on one line.

The last line, **Default Parameters**, is very useful and important. In this example, you would certainly not want to carry out a Quick Format on the default disk, which would normally be the hard disk C:. By specifying **A:** in this line, you ensure that careless or unthinking use of the Quick Format command will result in the disk in the A: drive being formatted rather than the hard disk.

- You will have to choose defaults that suit the program you want to run. In some cases, no default is really appropriate, and you would leave the line blank. Sometimes a default can be used to prevent a program from failing due to lack of a parameter; in other cases, as in this one, the default is a useful safety measure to ensure that you do not commit a terrible mistake by pressing the **ENTER** key without thinking.

On the **Commands** line of the Program Item Properties form you can list programs, including batch files, that need to be run both before and after the main program runs. This is the equivalent of batch file action, giving you all the powerful features of batch file use along with the easy and fast action of MS-DOS commands.

38 Creating a Group

The groups that exist when you first use DOSSHELL have been provided by Microsoft, and consist of **Main**, which contains the Command Prompt (temporary return to MS-DOS directly) and the **Disk Utilities Group**. Any other groups have to be created by you, using the procedures that will now be described. As with all other DOSSHELL actions of this type, you only have to specify what you want in a form rather than type an elaborate command.

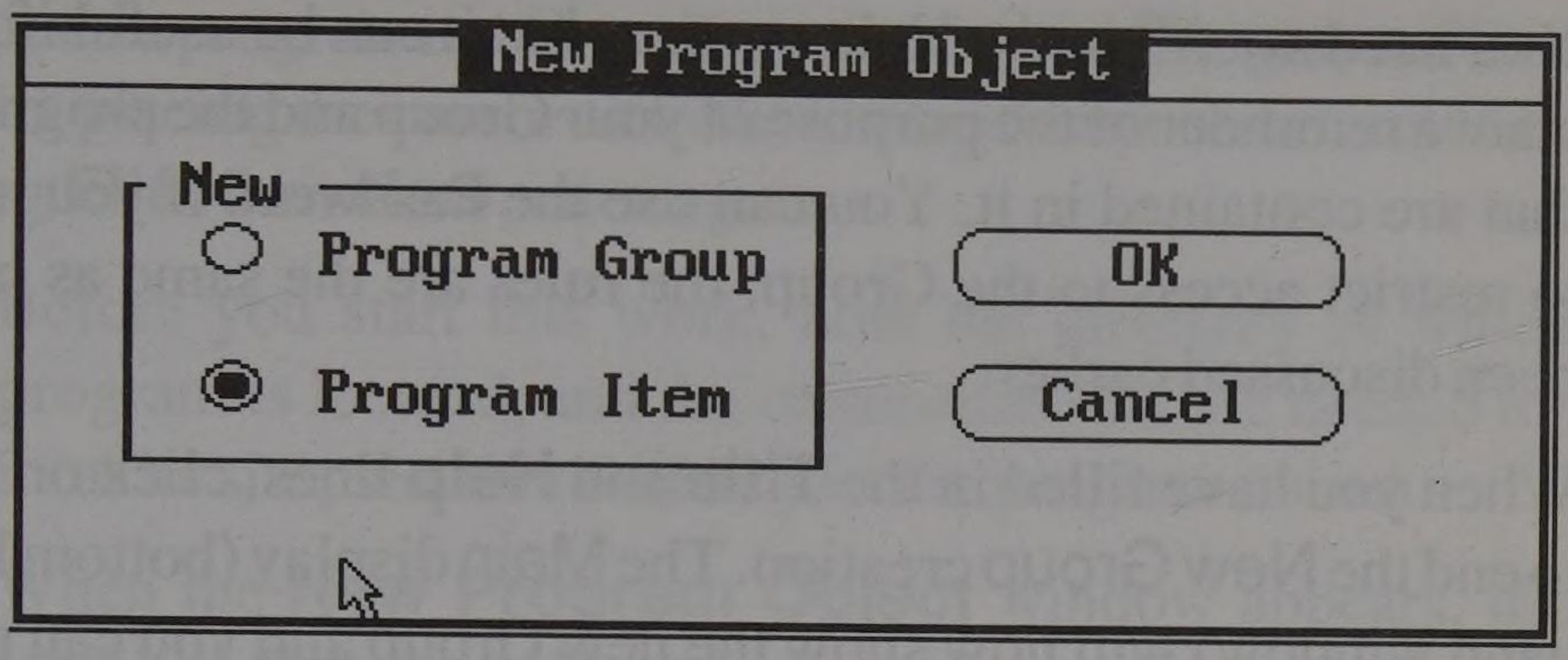
You are not forced to place programs into a Group when you create the Group, and in most cases you will add programs to a Group later, and not necessarily in one operation. You will often find that in the course of using a Group you want to add programs, and this is provided for in DOSSHELL.

The **Program Item Properties** item on the **File** menu also allows you to edit these properties at any time. In this way you can alter whatever you typed when you added a program to a Group. It is likely that through experience you will want to modify these Properties fairly often in the lifetime of a program.

The creation of a new Group starts with the display of Program files - the most convenient option from the **View** menu is **Program/File Lists**.

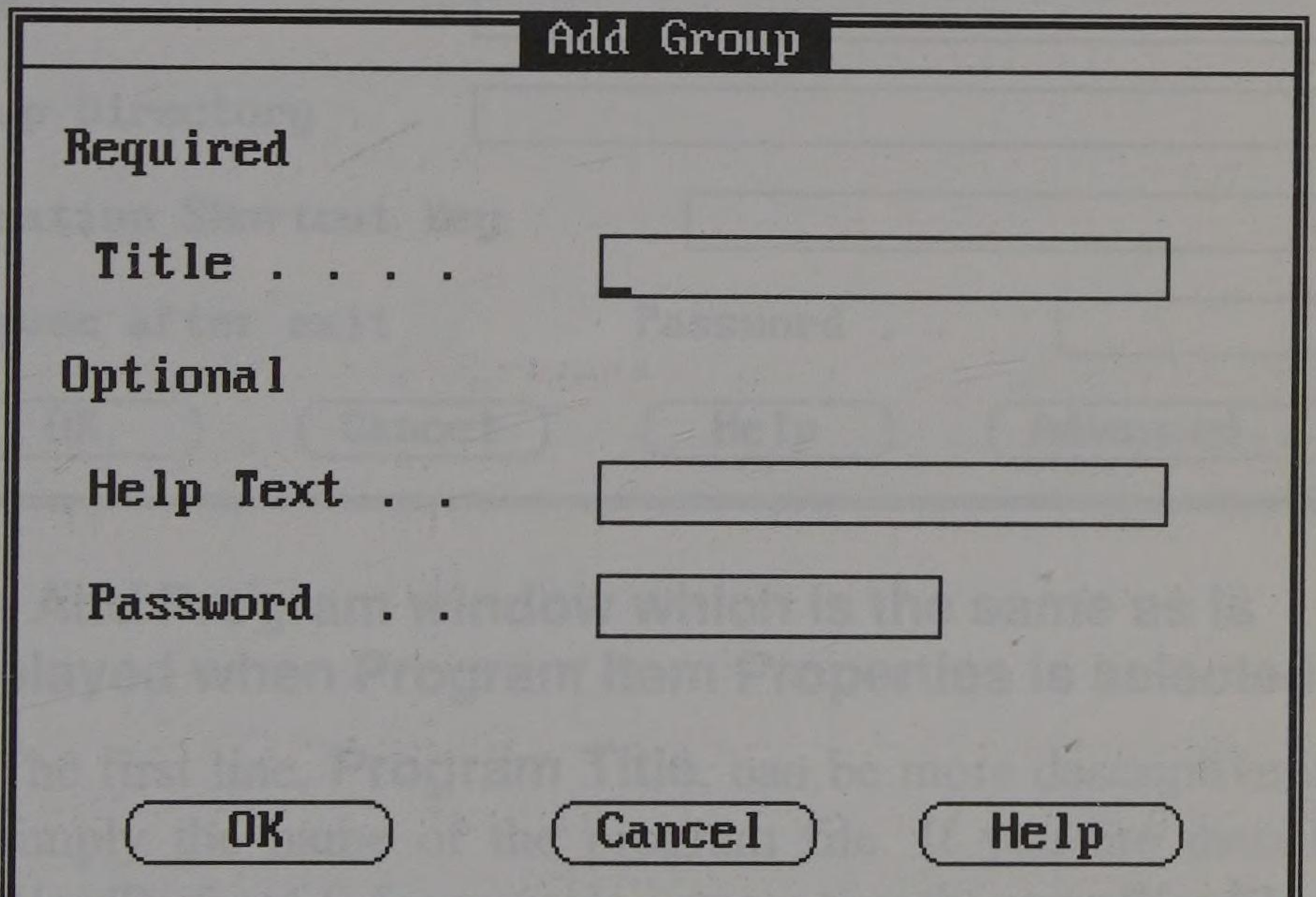
1. Place the cursor on the word **Main** in the Groups list, or on any of the names in the window that is headed **Main** so that one of the items in this set is highlighted. When you do this the **File** menu will contain the item **New** that allows you to create a new group or to add a program to an existing Group.
2. Selecting **New** brings up the **New Program Object** window, which allows you to choose **Program Group** or **Program Item** - the default is Program Item.

Creating a Group 38



The New Program Object window. The default is a new Program Item, but in this example we need to click on Program Group

3. Change the selection to **Program Group** and click on the **OK** box. You will now see the **Add Group** window appear. This allows you to type a title for the new Group, and to add the optional items of **Help** and **Password**.



The Add Group window which requires a title line - other lines are optional.

- You need not fill in the Help portion, but it can be useful if you want a reminder of the purpose of your Group and the programs that are contained in it. You can use the Password if you need to restrict access to the Group; the rules are the same as have been discussed earlier.

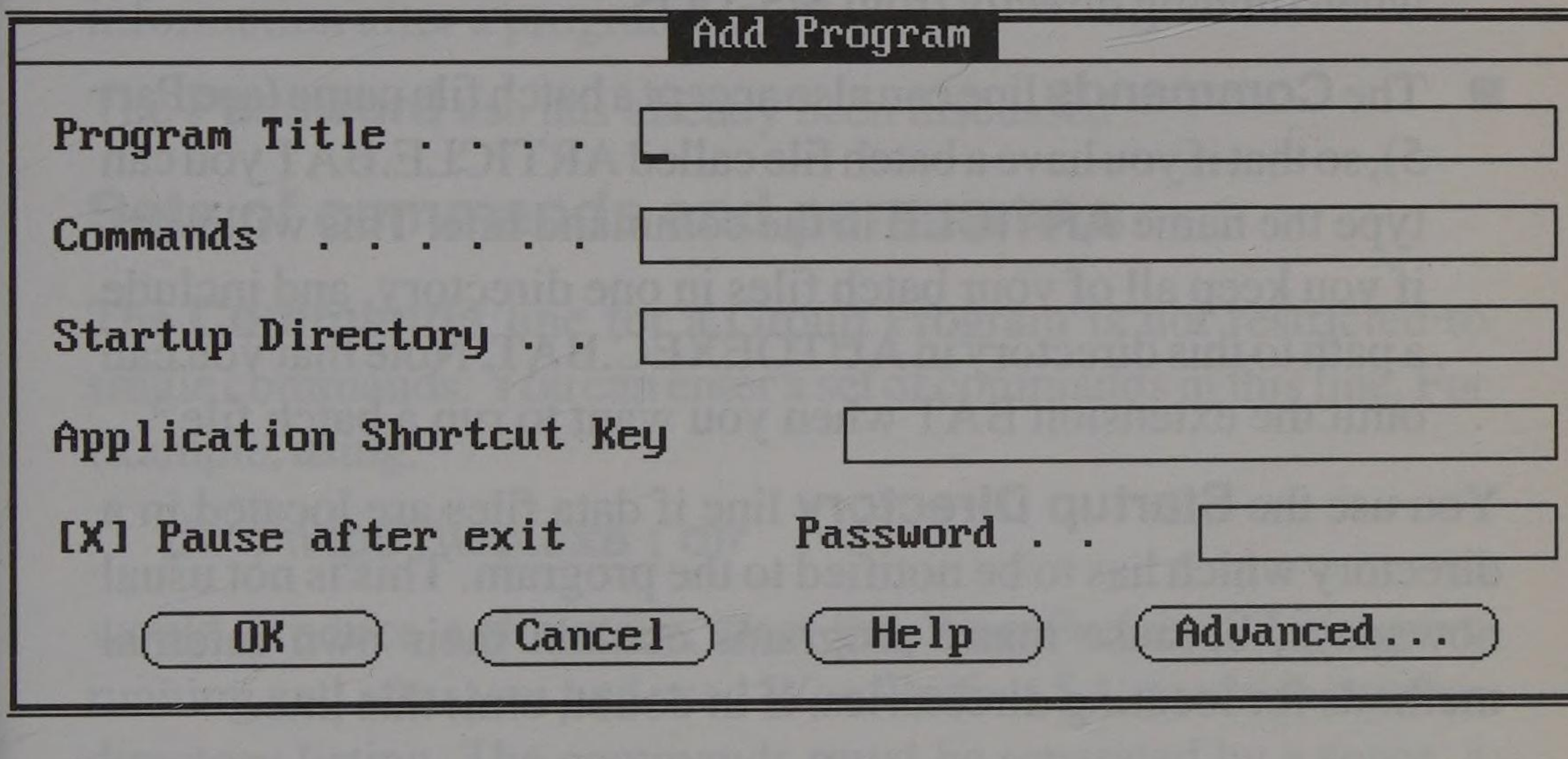
4. When you have filled in the **Title** and **Help** lines, click on **OK** to end the New Group creation. The Main display (bottom left-hand window) will now show the new Group and you can now place programs in it.

- Placing a program in a Group does **not** make a new copy of the program, it only provides access to the program.

Adding a Program to a Group 39

You add a program to a Group by selecting the **Program/File** view again, placing the cursor on to the Group that you have created, and using **File**, then **New**, once again.

1. Before you start this work, note the directory in which the program is located, and the commands that are needed to start it (at its simplest this will be just the program name).
2. When the **New Program Object** window appears, use the (default) **Program Item**.
3. When you click on **OK** you will see the **Add Program** window, which is identical in form to the **Program Item Properties** window on page 125.



The Add Program window which is the same as is displayed when Program Item Properties is selected.

4. The first line, **Program Title**, can be more descriptive than simply the name of the program file. If you are installing WordPerfect 6.0, for example, you can type the name *WordPerfect 6.0* in this line rather than the name of the main program file, which is WP.EXE.

- You can elaborate the name - you might want to have one WordPerfect just for letters and another for writing magazine articles for PC Answers, so you might have a **Program Title** line such as *WordPerfect for Letters* or *WordPerfect for PC Answers*. What you put here should remind you of what the program is and what it does.

5. The **Commands** line must contain the command that starts the program running. This might be a COM or EXE program file, in which case you would type the complete file name **with** the extension, not as you would when running the program from MS-DOS directly. For example, to run WP.EXE you would use **WP.EXE** as the command, rather than WP as you would use when running directly from MS-DOS.

- The **Commands** line can also accept a batch file name (see Part 5), so that if you have a batch file called ARTICLE.BAT you can type the name **ARTICLE** in the command line. This will work if you keep all of your batch files in one directory, and include a path to this directory in AUTOEXEC.BAT. Note that you can omit the extension BAT when you want to run a batch file.

You use the **Startup Directory** line if data files are located in a directory which has to be notified to the program. This is not usual nowadays, because many programs contain their own internal methods for locating directories. If in doubt, omit this line.

Specifying an **Application Shortcut Key** is not necessary unless you have planned to use this program as one you will be frequently swapping in and out of. It is usually better to leave this line vacant when you first place a program in a group, because you need to have your Group complete before you can plan what keys you will use. The normal entry will be something such as **ALT A**. (Press **Alt** key and **A** key; do not type the letters *ALT A*).

There are reserved combinations, which are not available as shortcut keys. These are:

Ctrl-C **Ctrl-H** **Ctrl-I**

Ctrl-MA **Ctrl-[** **Ctrl-5**

and the above set along with **SHIFT** (such as **Ctrl-SHIFT-C**). The **Ctrl-5** and **Ctrl-SHIFT-5** refer to the use of the **5** on the number-pad at the right of the keyboard, not the **5** above the letter keys.

The **Pause after Exit** box contains a cross by default, and you would normally remove this cross by clicking on it (click again to restore) to allow you to return to DOSSHELL automatically after a program has ended. Cross the box only if you need to read information after a program has run.

The **Password** use has already been discussed .

Sets of commands and parameters

The **Commands** line for a Group Program is not restricted to single commands. You can enter a set of commands in this line. For example, using:

dir ; \wp51\wp.exe ; dir

would produce a directory, start the WordPerfect 5.1 program running and after you had used WordPerfect 5.1, produce another directory listing. The commands **must** be separated by a space, a semicolon and another space - this is important. You can use more than one space, but you **must not** omit either space.

This structure allows the command line to produce the actions of a batch file, and you can even include **PAUSE** as a command to make sure that a directory listing is held on screen until you have had time to read it.

39 Adding a Program to a Group

The **Commands** line can contain parameters such as **%1, %2,** etc. in addition to the program names and batch commands. You type a parameter exactly as you would for a batch file (see Sections 48 - 50), that is with a space between the program name and the parameter. You can use up to nine parameters if you have any program that accepts such a large number.

A program called NOTE.EXE for example, might have been written so as to be used in the form NOTE MYNOTE.TXT so that a filename could be passed to it. You would place this in the command line in the form

NOTE.EXE %1

with a space between the program name and the parameter.

- When one or more parameters are included in a command line, the **Prompt Dialog** box will appear when you click on the **OK** box of the **Add Program** window.

You will be asked to supply, for each of the parameters, the information of **Window Title, Program Information, Prompt message** and **Default Parameters**. These pieces of information have already been mentioned, and for the example of a program called NOTE you would complete the form as follows:

1. The **Window Title** should remind you of the purpose of the program.
2. You can edit the **Program Information** line. In the example, you might type **File Name** in this line. This ensures that you see this heading when you start the program.
3. The **Prompt Message** is a further reminder. You are allowed up to 18 characters for a message such as: **Please type name.**

4. You can supply a **Default**, such as TEST.DOC, using the last line of the form. Supply a default only if it makes sense to do so; this is a very useful provision, but in many cases there is no need for any default.

5. Select the **OK** box on this form to end the work of creating a Program Item

Running a program from a group consists of two main stages followed by some subsidiary ones.

1. You must first select the Group by double-clicking on the **Group name** when a program display is available. This will list the programs in the Group, along with the Main Group name to allow return to the other Groups.
2. You then double-click on the name of the program in the Group that you want to run.
3. You will be shown any parameter entry form, in the style that you have set up when you filled in the Parameters form.

■ The prompts should help you to type the parameter that is required - you can, with some planning, ensure that you never at this point need to delve into the manuals for the program to find out what you need to type.

4. When you type one parameter and click on **OK** or press **ENTER**, you will see the next parameter entry box, if there is a second parameter, or see the program start. If no parameters are specified, the program will start when you double-click on its name, with no parameters box appearing at all.

Advanced Setup

When you first add a program to a Group, ignore the **Advanced** section, because it is not needed initially. The Advanced section is designed to cope with special situations, and the most useful part of it for most purposes is the Help Text provision - study the Help Text for the existing programs to see just what can be achieved.

The Advanced section is designed to cope with problems that are fairly unusual, and you need to know what the problems are and how to overcome them. You should not therefore make any use of the Advanced section (other than Help text) until you have considerably more experience of MS-DOS 6.0. In particular, do not change any of the memory size quantities unless you have been advised to do so,

Task Swapping with DOSSHELL 41

DOSSHELL allows you to swap from one program to another without the need to shut down a program first. The convenience of this is immense if you have never experienced it before. You could, for example, be working on a letter, using a word processor. At some stage, you need to look at a worksheet to find a figure from an accounts statement. The conventional method would be to save the word processor letter file, shut down the word processor, open up the spreadsheet, load in the correct worksheet, read the figures, jot them down, save the worksheet, quit the spreadsheet, open the word processor, find the letter file and load it in, and then find the correct position in the document.

Using DOSSHELL task swapping this can be reduced to pressing a key combination to save the word processor on disk, pressing another key combination to start the spreadsheet, finding the figures, and pressing another key combination to get the word processor back, still at the same place in the document. The very considerable advantage in this is that it allows you to **suspend**, rather than quit, a program, with the document or data file still in place and ready to resume. This avoids the most infuriating part of shutting down a program - the difficulty in finding the data file, loading it in again and finding the correct place to resume.

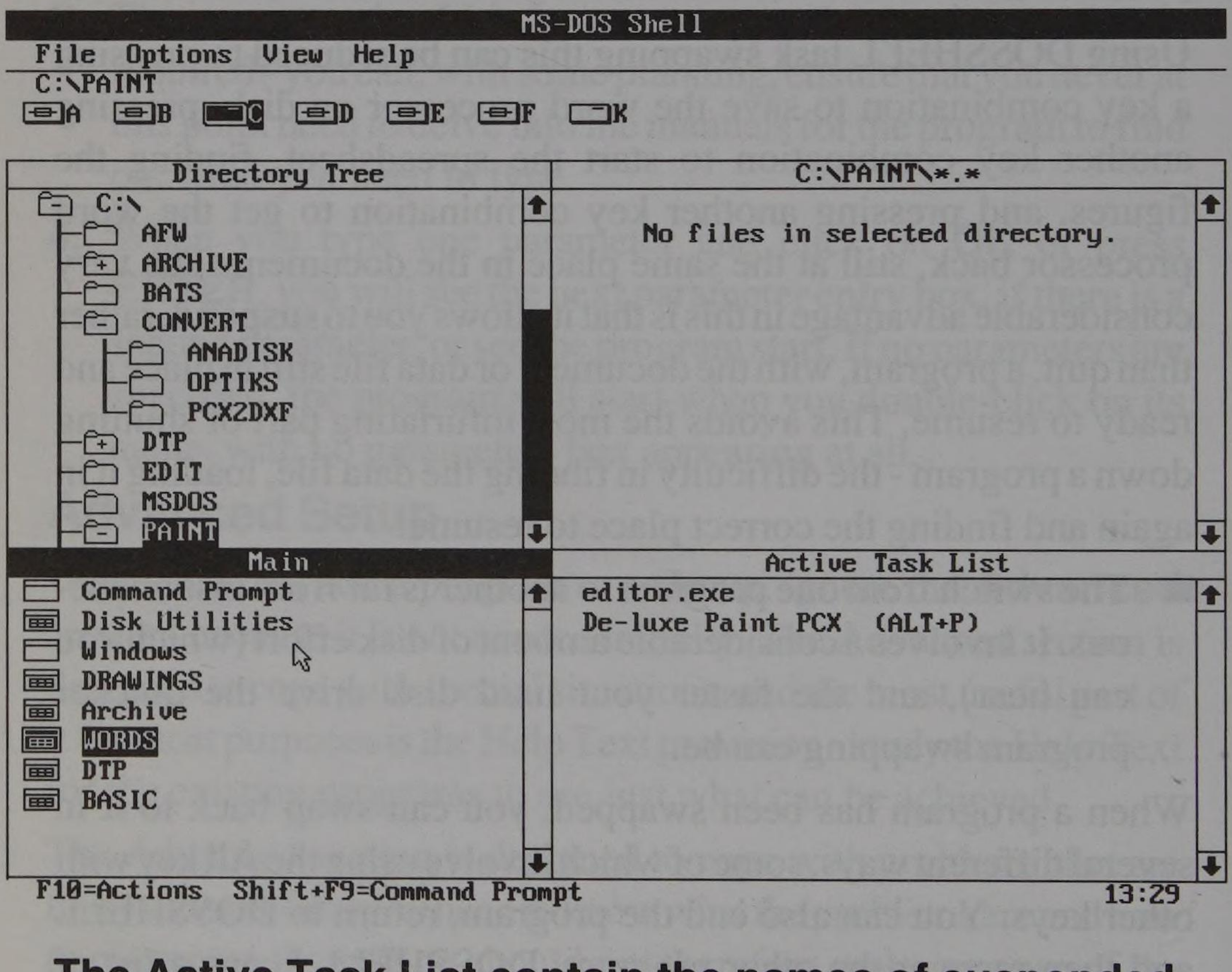
- The switch from one program to another is far from instantaneous. It involves a considerable amount of disk effort (which you can hear), and the faster your hard disk drive the quicker program swapping can be.

When a program has been swapped, you can swap back to it in several different ways, some of which involve using the **Alt** key with other keys. You can also end the program, return to DOSSHELL and then recover the other program. DOSSHELL keeps a list of programs that have been swapped out, the **Active Task List**, from which you can select a program to use by double-clicking on its name.

41 Task Swapping with DOSSHELL

■ The alternative is to assign to every program in a group a short-cut key, such as **Alt-A, Alt-B, Alt-C** and so on. This allows you to run these programs by using these key combinations, either from an existing program or from DOSSHELL.

No swapping is possible at all unless it has been enabled. To enable swapping, select the **Enable Task Swapper** line in the **Options** menu. A dot will appear against this line when swapping is enabled. If swapping is not enabled, none of the swapping commands will have any effect, and program names will not appear in the task-swapping list.



The Active Task List contain the names of suspended programs. In this pair, De-luxe Paint can be selected by pressing Alt-P or by double-clicking on the name.

Task Swapping with DOSSHELL 41

Assuming that swapping is enabled, there are several methods that you can use to start a program and to swap out of it. The most straightforward is simply to start the program running in the normal way, but you have the alternative of placing one or more program names in the Active Task List, without running any of the programs.

You can run any program whose name appears in the Active Task List by double-clicking on the name, just as if the name were in a Group list. Programs placed into the Active Task List need not all belong to one Group, but if they use Short-cut keys you need to be sure that no two use the same Short-cut key. You also need to be careful that you do not start more than one copy of the same program by forgetting that you have a program suspended.

Running and swapping

To use running and swapping:

1. Make sure that **Task Swapping** is enabled from the **Options** menu, so that the **Active Task List** (initially empty) appears.

2. Start and make use of the program in the usual way, but do not quit the program. Instead, use one of the following key combinations:

   Ctrl-Esc

   Alt-Tab

   Alt-Esc

   which, when you are starting a task list, will return you to DOSSHELL.

- The key combinations of **Alt-Tab** and **Alt-Esc** will have different effects when two or more programs exist in the Active Task List and you press the keys while you are using DOSSHELL.

3. While swapping is being done, the screen will be blanked out. Do not assume that something has gone wrong unless the blanking lasts for an unusually long time. This is done while the program is being stored on the hard disk. If you start a program and quit in the normal way by selecting a **Quit** option, the program is **not** put into the Active Task List, and you cannot swap directly between it and any other program.

- Some programs, such as large CAD and DTP programs will not swap because their memory requirements are too large and the portion of DOSSHELL that deals with swapping takes up enough space to prevent the program running.
- DOSSHELL allows you the option of creating your Active Task List **without** actually starting any of the programs. This is done by holding down the **SHIFT** key while you double-click on the program names. Each name will be added to the Active Task List and you can start it from the list by double-clicking on the name in the normal way.

When an Active Task List exists, you have several options for running programs in the list.

1. You can double-click on the program name in the usual way.
2. You can hold down the **Alt** key and press the **Tab** key. This will bring up a program name at the top of the screen and if this is not the program you want you can press the **Tab** key again, repeating until you see the program that you want. Release the **Alt** key to run this program.

- The **Alt-Tab** combination cycles from one program in the Active Task List to another. When you are using one of the programs in the Active Task List, using **Alt-Esc** will cycle to the next program in the list until the DOSSHELL appears.

3. The best method of program swapping is the use of **short-cut keys**, because this allows you to swap from any program to any other in the Active Task List without needing to cycle through the other programs in the List nor to return to the DOSSHELL.

- To make use of short-cut keys you must have designated a short-cut key for each program you are using (which implies that each program is one in a Group) and you must know what short-cut keys correspond to each program. The Active Task List will show the programs that have such keys assigned, and what keys are used.

Leaving a program

You can leave a program that is currently running by using its Quit or Exit option in the usual way. When you do this, the name will no longer appear on the Active Task List. Leaving other programs on the Active Task List must be done by running each program and using its Quit or Exit option.

- You must not simply exit from DOSSHELL leaving programs in limbo - you must make each program active and then quit it. If you simply switch off the computer you risk losing data from the programs in the Active Task List.

Unless you use the Quit option, you should be able to swap from one program to another indefinitely, using programs in any sequence. You can also return to DOSSHELL and add other programs to the Active Task List. If you make use of the **Command Prompt** from DOSSHELL to run MS-DOS, you can put MS-DOS into your Active Task List by using **Ctrl-Esc** (or any other of the set on page 143); but if you want to quit Command Prompt you must type **EXIT** and press **ENTER**.

41 Task Swapping with DOSSHELL

If a program locks up while being used with swapping, DOSSHELL provides for a way out. The **Ctrl-Esc** key combination should allow you to return to the DOSSHELL, and once back, you must quit the offending program by selecting its name on the Active File List and then pressing the **Del** key.

It is always better to restart the computer after a program has caused a lockup. This is the only exception to the rule that you always quit each program in turn, because attempting to use the other programs might cause a further lock-up. Be careful, in such an emergency, to remember to select the program name on the **Active Task List** and not in the **Files** list or the **Programs** list - using **Del** on these names would delete the programs from the disk.

DOSSHELL Options 42

The **Options** menu allows you a considerable choice, some of which concern display options and system options. The list presented in the Options set is:

Confirmation
File Display Options
Select across directories
Show information
Enable Task Swapper
Display
Colours

of which **Select across Directories** and **Show Information** do not lead to any other menu. The **Task Swapping** option has already been dealt with and none of the others is essential when you first start to use DOSSHELL.

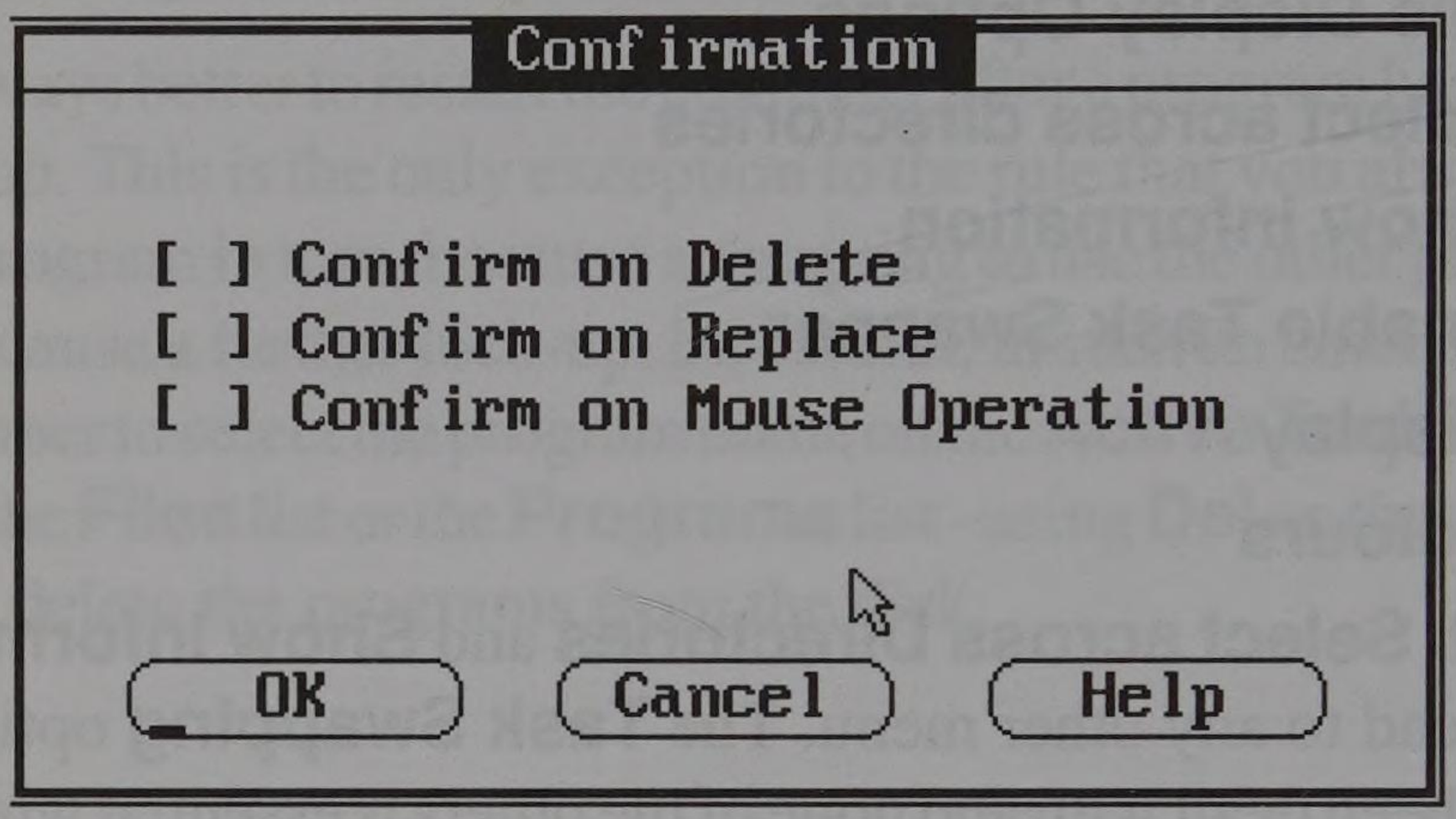
Confirmation options provide a safeguard against carrying out actions hastily, such as marking a set of files for deletion and then realising that some files ought to have been retained. This is particularly likely if you have selected a set of files by extension letters, without looking at the file names individually.

When you have switched on confirmation, you will be asked to confirm each action **individually**, so that whether you are working with one file or a large block, each name will be shown and you will have to reply by clicking on the appropriate box, **OK, Cancel** or **Help**.

There are three classes of confirmation actions, and by default all three are marked on. They are:

Confirm on Delete
Confirm on Replace
Confirm on Mouse Operation

Each has beside it a box that is either empty or marked with a cross. Clicking with the mouse cursor on the cross will remove the cross, also removing the confirmation action. Clicking on an empty box will place a cross in the box, re-establishing the confirmation action.



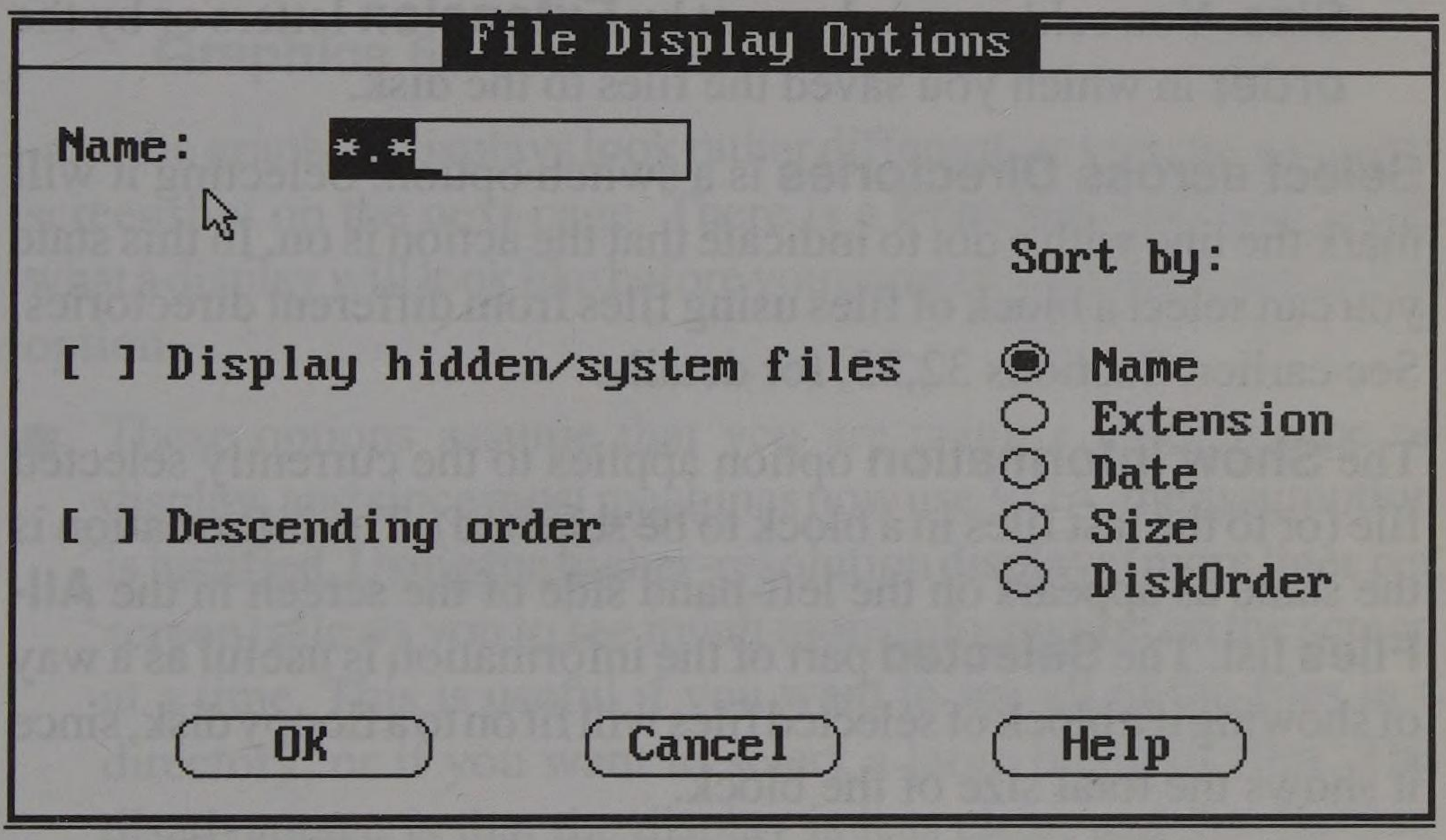
The Confirmation options list. To turn a confirmation on click on the box so that a cross appears.

1. Use **Confirm on Delete** if you are worried about accidentally deleting files. Once you are experienced with the use of DOSSHELL, it can be useful to remove this safeguard, because it can make a long task of deleting a large bunch of files. With this confirmation switched off, you can mark a block of files, press the **Delete** key, confirm once and delete all of the files with no additional confirmation for each individual filename.

2. Use **Confirm on Replace** as a safeguard when you copy files. If a file of the same name exists in the destination disk or directory, this option will require you to confirm that the new file is to replace the old file, and to assist you, the details of each file are shown. This also is a useful safeguard when you start to use DOSSHELL at first, but you may want to dispense with it later, particularly if you make a lot of backups of files.

3. Use **Confirm on Mouse Operation** when you carry out block actions on files using the mouse rather than by use of the keyboard or menu commands. It is remarkably easy, when copying a set of files to a directory using the mouse, to make the copy to the wrong directory, and by retaining the cross in this option box, you will be reminded by a message of what you are trying to do and be given an opportunity to escape or confirm.

- In general, you should keep the Confirm options switched on unless you find them restrictive. You can always switch a Confirm option off while you carry out a block action, and then switch it on again afterwards. Once you have some experience in using DOSSHELL you may want to turn off one or more of the Confirmation actions.



The File Display options that offer the options of showing Hidden or System files, sorting into descending order, and all the sort options

42 DOSSHELL Options

When you click on **File Display Options**, you will see the Form appear. You can use the options as follows:

1. Use the **Name** space to type a wildcard description such as *.TXT or MYFILE.*
2. Leave the **Display hidden/system files** box uncrossed. You need this only for advanced work, and keeping these files hidden is a safeguard against deleting essential files and losing control of the computer.
3. Click on **Descending order** only if you want to see files sorted in Z to A order.
4. The other options are for sorting order. The default is to sort by **Name**, but for some purposes it is useful to sort by **Date** or **Size**. You seldom need to sort by **Extension** letters or by the **order** in which you saved the files to the disk.

Select across Directories is a switch option. Selecting it will mark the line with a dot to indicate that the action is on. In this state you can select a block of files using files from different directories. See earlier, Sections 32,33, for details.

The **Show Information** option applies to the currently selected file (or to the last files in a block to be selected). The information is the same as appears on the left-hand side of the screen in the **All-Files** list. The **Selected** part of the information is useful as a way of showing if a block of selected files will fit on to a floppy disk, since it shows the total size of the block.

Display Options 43

You can use the display options to determine whether you use the text or the graphics screen, and the number of lines of information that can be placed on the screen. When this option is selected, the menu that appears offers the choice, for VGA screens, of:

Text 25 lines

Text 43 lines

Text 50 lines

Graphics 25 lines

Graphics 30 lines

Graphics 34 lines

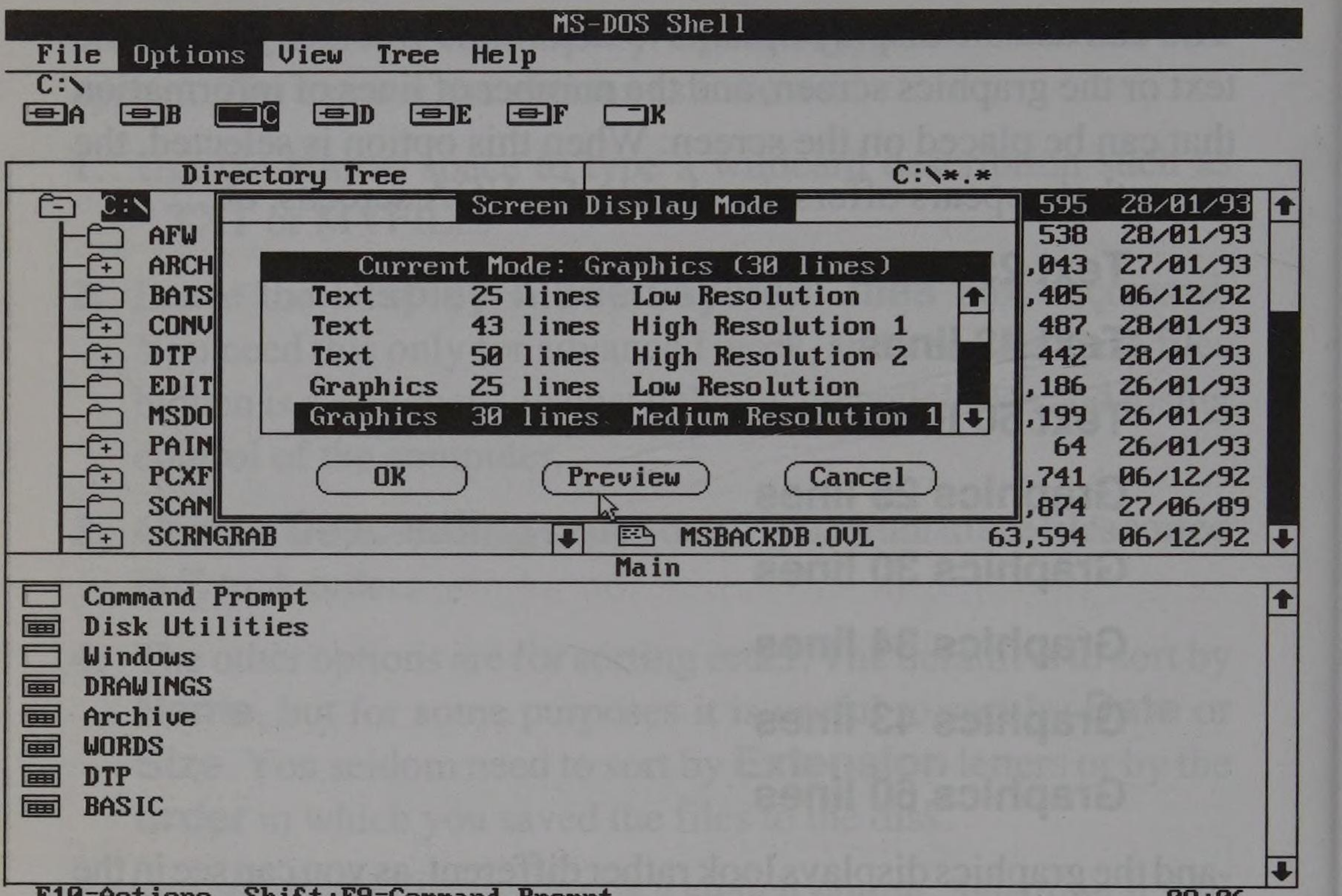
Graphics 43 lines

Graphics 60 lines

-and the graphics displays look rather different, as you can see in the screenshot on the next page. There is a **Preview** option to show what a display will look like before you commit yourself to using that option.

- These options assume that you are using a modern type of display, and since most machines now use VGA, the assumption is justified. Using the higher-resolution displays (more lines per screen) allows you to see much more information on the screen at a time. This is useful if you want to see all of the files in a directory, or if you want to select a large block of files. The disadvantage is that the display is less clear, and you have to move the mouse very carefully if you want to select one file out of the list.

43 Display Options



The typical Graphics screen appearance of DOSSHELL, with the Screen Display Mode panel superimposed. The Preview option allows you to see what a display mode looks like.

The **Color options** are not particularly relevant if you are using a monochrome display, but they allow a considerable choice for users of VGA colour screens.

The range of options is:

Basic Blue
Monochrome-2
Reverse
Emerald City
Ocean
Monochrome-4
Hot Pink
Turquoise

and if you are using a monochrome VGA display (which is much sharper and clearer than a colour display) the **Monochrome-4** option is a very good one, allowing for two shades of grey along with black and white. If you have a monochrome monitor you will find only the Monochrome options available - the colour options used on a mono monitor look the same as the Monochrome-4 option.

As far as the colour displays are concerned, if you are using a colour monitor you can please yourself - some combinations are easier to read than others but all are usable. Remember that if you want to print screens the printing will usually be in black and white and in some cases it is better to use the Monochrome-4 option even with a colour VGA screen in order to see what will be printed.

IMPORTANT NOTE: Some MS-DOS utilities require you to leave DOSSHELL or Windows before you start the utility. It is very important to make sure that you have completely ended use of DOSSHELL or Windows, with no programs suspended, when you use such utilities. Failure to do this can result in loss of data. The types of programs that require you to leave DOSSHELL or Windows are dealt with in Part 6.

- 5 -

Deeper into DOS

44 Commands and DOSKEY

When you have typed and used a command in MS-DOS 6.0 you might think that you have lost it for ever. It is not lost, and if you want to repeat a command you need only press the **F3** key to display the whole of the most recent command. You can also use the **F1** key to repeat the most recent command character by character. These Function Key uses can be extended by a program called **DOSKEY** that is normally run from the AUTOEXEC.BAT file, see Section 18.

- Once **DOSKEY** has been run, all commands are stored (up to a default limit of 1024 characters - more can be obtained) and can henceforth be recalled in order by using the cursor **Up/Down** keys and the **Page Up** and **Page Down** keys.

Suppose, for example that after starting the machine you used **DIR**, then **CD\MSDOS** and **DIR/W**, followed by **COPY *.SYS B:**. After using this set, the **Page Down** key will recall the most recent command (the **COPY** command) and **Page Up** will recall the oldest, the **DIR** command. The **Up** arrow will recall an earlier command than the current one, and the **Down** arrow key will move to a later command. You can run any recalled command, as normal, by using the **ENTER** key.

DOSKEY also allows you to edit commands. Suppose, for example, you typed as a command line:

PINTIT HORSE.DWG

intending to type **PAINTIT** as the name of a program. After running DOSKEY you can use the cursor keys to move the position for editing (used alone to move by one character, with **Ctrl** to move by one word) or use the **HOME** and **END** keys to move to the start or the end of the line respectively. This makes it easy to move back to put in the letter **A**. This editing of a command line **cannot** be done unless DOSKEY has been run.

- The default editing mode is **overtype**, meaning that a new character will replace the character under the cursor. This is nearly always the wrong default for users, who would prefer to be able to insert a letter, as in this example.

- If you want to use **Insert** as a default, so that a new character will be inserted at the cursor position, install DOSKEY in AUTOEXEC.BAT by using the line:

  DOSKEY /INSERT

You can use the normal delete keys on any line, and pressing **Esc** will delete the whole command.

The other options are **/reinstall**, **/bufsize**, **/M** (or **/MACROS**), and **/H** (or **/HISTORY**). It is unlikely that you will need any of these until you have considerably more experience with DOSKEY.

The Function key actions when DOSKEY is in use are.

Key	Action
F1	Copies characters from the stored line to the screen, one by one each time the key is pressed.
F2	Searches for the next key character (so that **F2 X** will find an **X** in the stored line) and insert all of the stored line up to that point
F3	Copies the whole of the stored line to the screen, or the remainder of the line if some has already been copied with **F1**.
F4	Searches for the next key you press (as for **F2**) and deletes up to that point in the stored line.
F5	Copies the visible command into store, and deletes the visible command (use **F3** to restore).
F6	Places the **Ctrl-Z** character, shown as **^Z**, at the end of the visible command line.

F7 Display all stored command lines, numbered.

F8 Search for a matching line in store type the first part of the command you want to recall and press F8 to get the full command

F10 Display all macros

You can also use the following **Alt F**-key combinations:

Alt-F7 Erase all stored command lines.

Alt-F10 Clear all macros

- You can also use Macro commands along with MS-DOS 6.0. These are sets of commands that are carried out when some key combination is pressed. Macro command programming is definitely not for newcomers, and when you have sufficient experience to benefit from Macros you will find information in the *Pocket Book of MS-DOS 6.0*. You can consider yourself sufficiently experienced when you have written batch files (see later this Part) for yourself.

Using DIRCMD 45

DIRCMD is a way of altering the way that **DIR** displays a directory. The default is to use a 40-column display, so that there is one file to a line. This can make it difficult to see what you want in a crowded directory, so that by using DIRCMD in your AUTOEXEC.BAT file you can tailor the DIR display to whatever you want.

The form of this line is:

SET DIRCMD=

with some letter codes following the equals sign. There must be **no** space on either side of the equals sign. **DIRCMD** is **not** a program name, it is a **variable** (technically, an environmental variable) meaning that it is a name that MS-DOS 6.0 uses to indicate a set of codes.

The codes that can be used following **SET DIRCMD=** are:

/a list only files with specified attributes, which are -

- **:a** files ready to backup
- **:-a** files unaltered since previous backup
- **:d** directories included
- **:-d** files only, no directories
- **:h** hidden files
- **:-h** excluding hidden files
- **:r** read-only files
- **:-r** excluding read-only files
- **:s** system files
- **:-s** excluding system files

/o control order of sorting displayed files:

:c sorted by compression ratio, lowest first

:-c sorted by compression ratio, highest first

:d sorted by date and time, earliest first

:-d sorted by date and time, most recent first

:e alphabetical order of extension letters

:-e reverse alphabetical order of extension letters

:g directories arranged before files

:-g directories arranged following files

:n alphabetical order of name

:-n reverse alphabetical order of name

:s order if file size, smallest first

:-s order of file size, largest first

/p one screen of listing at a time

/w list over the full width of the screen

/-w use only half the screen width for list

The main commands always start with the slash sign and where a command such as **/a** or **/o** can take other parameters (more than one if required) you can group these together, with no spaces, following one single colon.

For example, the line:

set dircmd=/w/p/o:gn

sets the directory listing as wide (**/w**), one screen at a time with no scrolling (**/p**) and with sorting in use with directories ahead of files and all in alphabetical order of name(**/o:gn**).

Using **SET DIRCMD** like this only sets the **default** format, and you can still obtain **DIR** listings in any form you want out of the possible set listed above.

For example, if the **DIRCMD** uses **/w** for a wide listing, which provides less detail about files, you can obtain an ordinary width listing by typing **DIR /-w**, turning off the **DIRCMD** setting temporarily for this command only.

46 Redirection

Redirection means transferring data from one output or input device to an alternative. The data will normally be the output from a program that would normally be directed to the screen, or the input to a program that would normally be taken from the keyboard .

- For example, using **DIR** will produce an output to the screen, and you might want to print the output on paper, or to place it in a file. This requires redirection of the output of the **DIR** command.

Redirection in MSDOS makes use of the signs **>** and **<** to indicate the direction of the flow of data, with **>** meaning a redirection of **output** and **<** meaning a redirection of **input**. Looking at output redirection first, the form of the command is :

command > output device

which makes use of the usual abbreviations for the output devices. These abbreviations are:

AUX The COM1 serial port (in or out)

CON Screen (output) or keyboard (input)

COM Serial port (in or out). You can use COM2, COM3 or COM4 as well as COM1

NUL Nothing, this will have no effect.

PRN Printer

For redirection to a file, the file path and name have to be specified in the usual way, such as **C:\NEWFILE\REDIR.TXT**.

The most common requirement is redirection of the screen output to a file or to a printer. For a file redirection, you will have to specify the filename. For redirection to a parallel printer, use **PRN**. There **must** be a space on the pointed side of the **>** sign.

Example: **DIR C:\wp\text > dirfil.doc**

will put the directory of **C:\wp\text** into a file called **dirfil.doc**. You can then use this file like any ASCII file, so that it can be read into a word processor, etc.

Example: **TYPE B:blurb.txt > prn**

will print the file **blurb.txt** on the parallel printer (the printer must be connected and switched on).

Example: **TYPE A:other.doc> aux**

will send the file **other.doc** to the serial port, to a serial printer or another computer. The serial port must have been correctly set up using **MODE**. The use of AUX and COM1 is interchangeable (you might have to specify COM2 if this is the serial port you use). This redirection works satisfactorily both with serial printers and with direct links to other computers, rather like using **COPY *filename* AUX**.

- Using the double redirection sign **>>** has the effect of **appending** data to an existing text file.

  Example: **DIR B:>> prnfil.doc**

  will **add** the directory of the disk in the B: drive to the file **prnfil.doc** (in the current drive) without deleting the present contents of the file. If no such file exists it will be created.

- This makes the use of **>>** safer, in the sense that it copes equally well with the possibilities of no file being ready; or a file existing with data you want to keep.

Another device name that can be used for redirection is NUL. When this is used, the output of the program disappears. It is neither displayed on screen nor printed. This can be used as a way of testing that something will be read correctly, and also as a way of making

sure that an output from an action that is carried out in a batch file does not cause any screen output (as it would even with **ECHO OFF** used).

Example: **DIR A:> NUL**

will read the directory of the disk in drive A and consign it to oblivion. The disk drive motor is the only indication that something has happened. This redirection is a useful way of preventing your screen from filling with unwanted messages while a program is working- it is particularly useful for the AUTOEXEC.BAT file because many of the programs that are installed memory-resident will provide messages which you will want to see only if you are making changes or in the event of problems.

For example, using, in AUTOEXEC.BAT, the line:

MOUSE > NUL

will activate the mouse, but prevent any messages about it from appearing on the screen. It's the mouse you want, not the messages.

We can seldom use input redirection in such straightforward ways, because it is seldom necessary to redirect an input and few programs take input in a way that permits redirection.

A **pipe** is a method of transferring data from one program to another, as distinct from transferring data from a program to a file or a device. The symbol we use for this action is the | sign between the programs that are to be connected in this way.

- You cannot assume that a program will accept input in this way. The only programs that you can use with certainty to accept piped input are the MSDOS commands or other programs written for the purpose. These latter programs are the **filters**, and their purpose is to alter the output from or the input to a program.

Piping is achieved by creating a temporary disk file, so that you cannot operate piping unless this file can be written. This is not possible if your disks are all writeprotected or if they are full. For example, if you type:

TYPE A:README.DOC | MORE

this will filter the output of the **TYPE** command through the **MORE** filter program. This, however, can work only if the MORE filter can create a temporary file on Drive A:, and this may not be possible if the disk is writeprotected or full. MSDOS 6.0 allows you to use a **TEMP** command in the AUTOEXEC.BAT file which provides a space on the hard disk (or another disk) for such files. This is done using:

SET TEMP=C:\temp

assuming that the directory called **temp** exists on Drive C in this example. You may have such a line in your AUTOEXEC.BAT file already for the purposes of another program, in which case MSDOS 6.0 will use the same directory when a pipe command is running. The temporary file that is created will be deleted after the pipe action has been completed.

It is important to have the **SET TEMP** line in your AUTOEXEC.BAT for many other purposes, and some program will put this line in for you if it does not already exist.

Filters

The filter programs that are supplied with MS-DOS 6.0 are **FIND**, **MORE** and **SORT**, of which you can use FIND for purposes that are separate from piping. When you install MS-DOS 6.0, **SETUP** will place these filter programs in the MSDOS directory, and the **PATH** line in the AUTOEXEC.BAT file will ensure that they can be called into use from any other directory.

The effect of a filter is to select or rearrange data that has been piped to it by a program, and then pass that modified data to the screen, or on to another program by way of a pipe, or to a printer by way of a redirection. Most filter programs are used in conjunction with **TYPE** or **DIR**, the two MSDOS commands that provide an output of text.

FIND

FIND is a good example to take to show the difference between program use and filter use for achieving the same purpose.

1. FIND "Spotless" textfil1.txt

2. TYPE textfil1.txt | FIND "Spotless"

The first example uses FIND as a program. This example will search the specified file for the word "**Spotless**" and place its output on the screen. The second example uses TYPE to pass text to FIND **used as a filter**, with the result again appearing on the screen. When you use FIND as a program, it has to be followed by the phrase that you want to find and also the filename.

Used as a filter, FIND appears following the pipe sign |, and is itself followed only by the phrase that is to be found. The text that the FIND filter is to search has to be passed along the pipe from another program, and in this case TYPE has been used as a way of selecting the text. Normally, TYPE would display the text on the screen, but with FIND interposed between the TYPE command and the screen output, only the selected output appears.

The use of FIND as a filter makes it of considerably wider use than FIND used only as a program. For example, suppose that the root directory of your hard disk contains several hundred files (which is bad organisation, but that's another matter). To find files created in 1991 we can use:

DIR C: | FIND "91"

because each date is stored in the form DDMMYY (unless you have forgotten to use the COUNTRY file to adapt to UK date conventions).

FIND Options

FIND allows four option letters **/C**, **/I**, **/N**, and **/V** that are placed between **FIND** and the item to be found.

/C alters the output of **FIND** to provide a count of the number of lines in which the item is found.

/I makes the action case-insensitive, so that a search for **Smith** will also produce **smith**.

/N prints out a line number as well as the line that contains the item.

/V displays the lines that do **not** contain the search item.

MORE

The **MORE** filter simply cuts text into pages that can be displayed on the screen. If you send data to the screen through the MORE filter, then one screen full of data will be displayed at a time, with the message

-- More --

appearing at the bottom of the screen. You can then press any key (use the spacebar preferably) to display the next page. To send a text file through this filter, you have to specify that the contents of the file will be piped to the filter. If the command that you use to provide the text normally produces its output on the screen, then the output of MORE is also sent to the screen.

Example: **TYPE READ.ME | MORE**

will place the contents of the file **READ.ME** on the screen, one page at a time, assuming that the **MORE** program is on the same disk as **READ.ME**.

You could, of course, further redirect the output to the printer by using the command in the form:

TYPE READ.ME | MORE > PRN

and an alternative is:

MORE < READ.ME > PRN

which uses redirection rather than piping.

- Note the different order of the commands, also that the **TYPE** command is **not** needed because the **<** sign implies that the file **READ.ME** is used as an input to **MORE**.

SORT

SORT is the other standard filter, and its effect is to arrange text data into alphabetical order of the letter in the first column (and when two words have the same first letter, then the second letter will be taken into account and so on).

Example: **DIR | SORT**

will place on to the screen a directory of the current disk or directory that is sorted in alphabetical order. The output of **DIR** has in this example been piped through **SORT**, and so to the screen.

Example: **TYPE INXIT | SORT**

when **INXIT** consists of a list of names in any order, will produce a sorted list on the screen.

The SORT command and filter allow options that can be a number or the letter R, placed immediately following SORT and separated by the usual stroke mark.

- You can use a **/+*number*** option, such as **/+5**, to sort the file by the order of the character in the **5**th column, in this example. You can use any valid column number, and if you omit this option, the first column will be used as a default. This can be useful if the list consists of four places for numbers, a letter, a comma, and then a word. The **+** sign must **not** be omitted.

The **/R** option sorts the items in reverse order, highest value first and then to lowest. If you need to use both **/R** and **/+*number*** options, the **/R** option **must** be placed first following **SORT**.

- SORT treats upper and lower case characters as identical, so that **c** and **C** are treated as the same character. You cannot use SORT on files of more than 64 Kbyte size.

48 Introducing Batch Files

Many users of the PC have only ever used the computer MS-DOS commands directly, typing a command and then pressing the **ENTER** key so that the computer carries out the command at once.

- This type of use is called **interactive**, and the alternative is called **batch** use, in which you issue a set of commands and then let the computer get on with the job of carrying out each one in sequence.

Obviously, you have to type the commands in rather a different way so that you can achieve this, because what you are doing is to write a miniature program of commands that the machine will execute later and in a particular order. This is done in MSDOS, much as it is done in other operating systems, by writing a file that is recorded on to a disk.

This type of file as used in the PC is called a **batch file**, and by typing the name of such a batch file, just as you type the name of a program, you can make the computer carry out, in sequence, the commands that are contained in the file. Each batch file has the extension letters of **BAT**, and you should not use these extension letters for any other type of file.

Suppose, for example, you created a batch file that contained instructions for copying all the files from the C:\WRITEM directory that you had modified since 1/1/1993 to a floppy in drive A. This batch file might be called BAKEMUP.BAT. The important point about such a batch file is that to carry out this action all you need to do is to place the disk in the drive and type **BAKEMUP**, the name of the batch file. This assumes that you have stored your batch files in a directory that is notified in the PATH command in AUTOEXEC.BAT, so that you can call up a batch file from any directory.

Introducing Batch Files 48

- Batch files provide a way of automating your work, so that you can achieve the effect of typing a number of commands by simply typing a single command, which is the name of a batch file.
- Batch files work particularly well in conjunction with DOSSHELL, allowing you a degree of control over your computer that is hard to achieve in other ways.

Creating a batch file

A batch file, at its simplest, is a set of normal MS-DOS commands that you type and record in ASCII file codes like any other text. You can create batch files with the EDIT editing program that is part of the MS-DOS 6.0 system when it has been correctly installed.

- You **cannot** prepare batch files with a word-processor unless you can set the word-processor to produce ASCII files - any other setting is likely to place unrecognisable codes into the text.
- The EDIT program is part of the language, QBASIC, that is supplied along with MS-DOS 6.0. If you delete the files of QBASIC in order to save disk space, you will be unable to use EDIT.

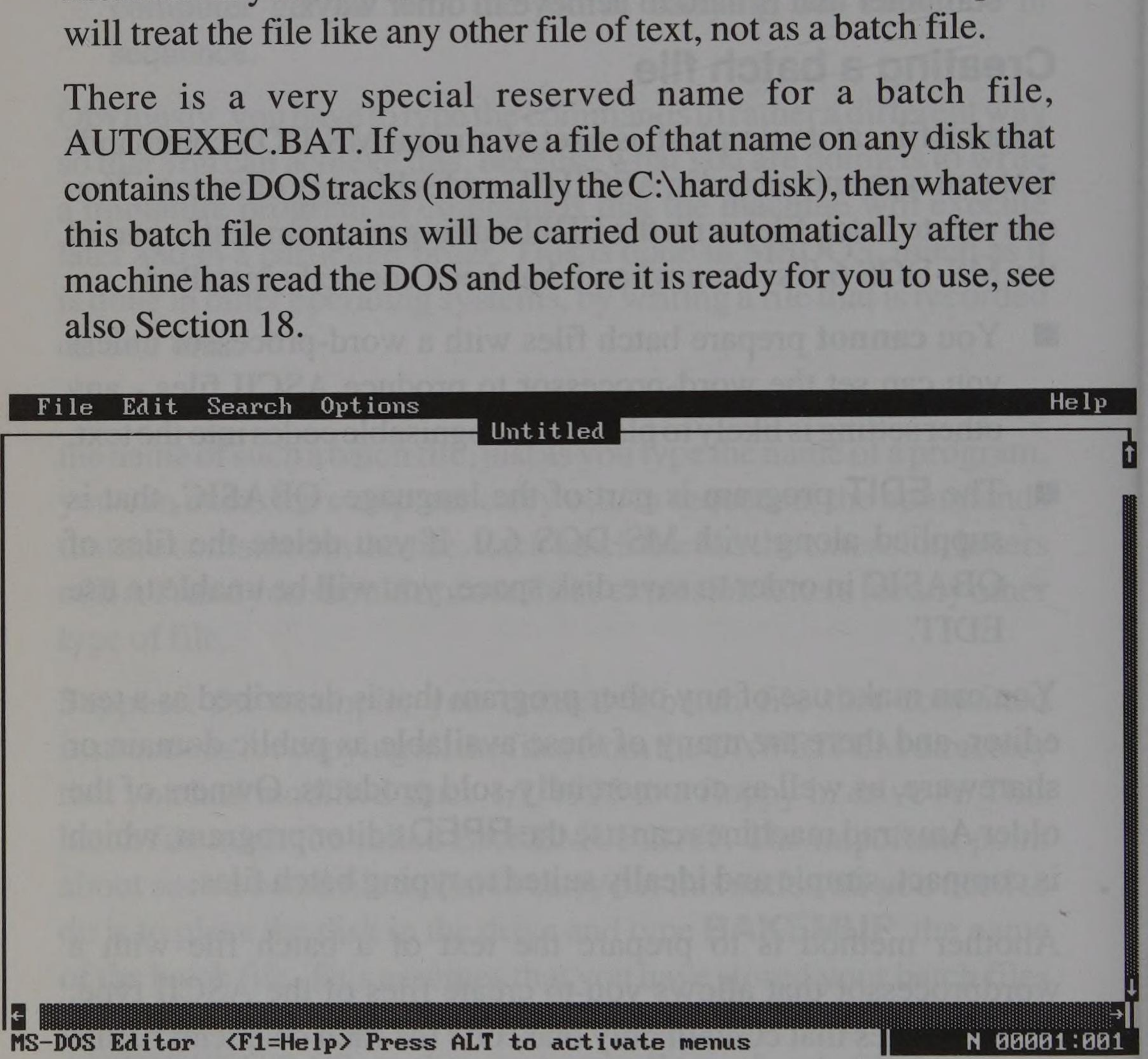
You can make use of any other program that is described as a text editor, and there are many of these available as public domain or shareware, as well as commercially-sold products. Owners of the older Amstrad machines can use the RPED editor program, which is compact, simple and ideally suited to typing batch files.

Another method is to prepare the text of a batch file with a wordprocessor that allows you to create files of the ASCII type, meaning files that contain only one code for each character on the keyboard, with no hidden codes for controlling printers.

A batch file, whatever its purpose, consists of the usual commands of MSDOS, plus a few very useful commands that are peculiar to

batch files, and which make the batch file much more like a programming language in its own right. Each batch file **must** have the extension letters BAT, since this identifies it as a batch file when you look at the disk directory. The fact that the file has this extension also allows the machine to locate the file when you type the main filename. If you do not use the extension BAT, then the computer will treat the file like any other file of text, not as a batch file.

There is a very special reserved name for a batch file, AUTOEXEC.BAT. If you have a file of that name on any disk that contains the DOS tracks (normally the C:\hard disk), then whatever this batch file contains will be carried out automatically after the machine has read the DOS and before it is ready for you to use, see also Section 18.



The initial appearance of the EDIT screen. Click on File to see the OPEN option.

Introducing Batch Files 48

The simple use of the EDIT program was covered in Section 17. The only commands you need to know for writing batch files are the **OPEN, SAVE** and **SAVE AS** commands, all of which are in the **File** menu.

OPEN is used to alter an existing batch file, and when you use OPEN you will be asked to supply the filename of the file you want to alter. EDIT assumes this will use the TXT extension, but all batch files use the BAT extension.

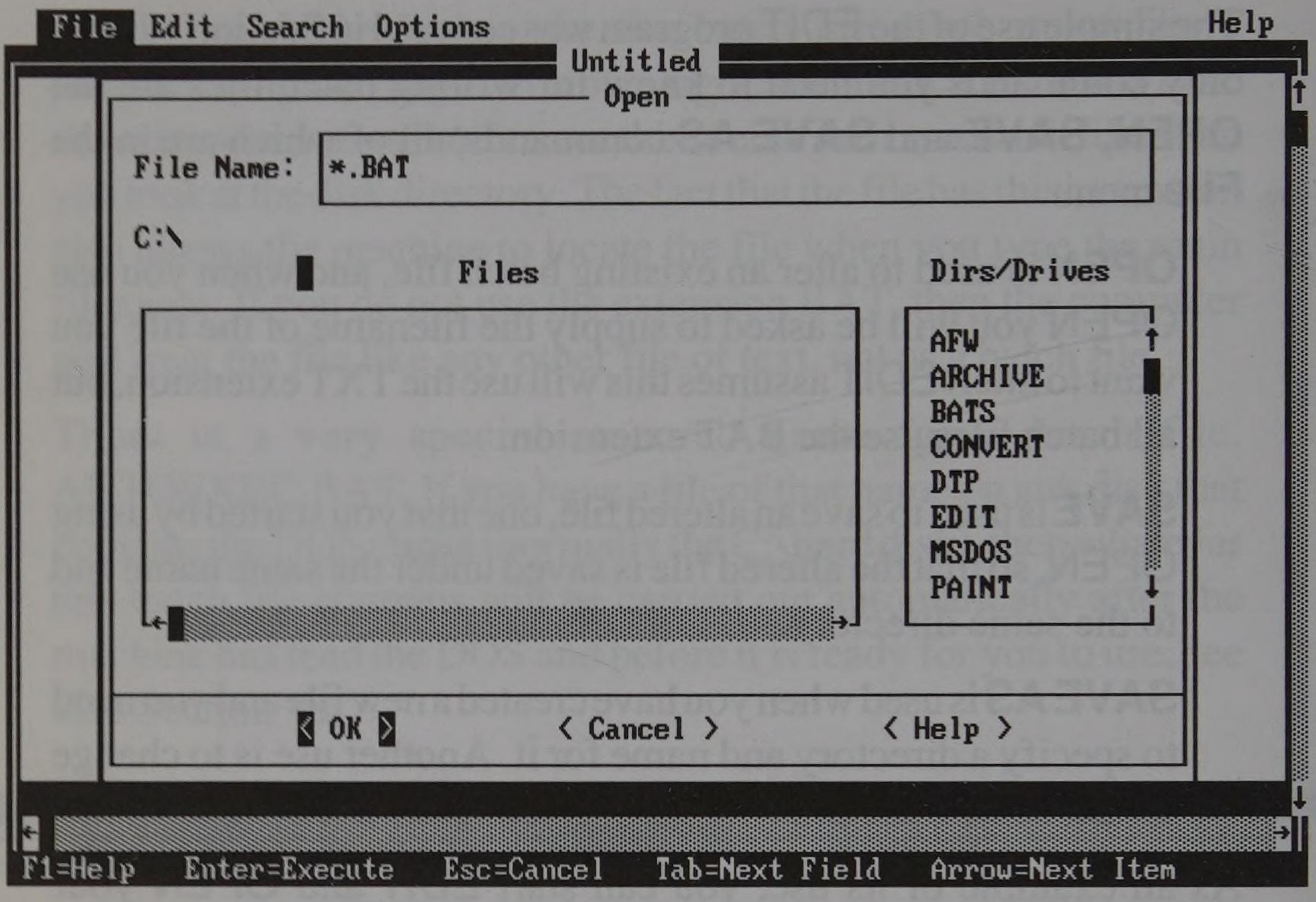
SAVE is used to save an altered file, one that you started by using OPEN, so that the altered file is saved under the same name and to the same directory.

SAVE AS is used when you have created a new file and you need to specify a directory and name for it. Another use is to change the name of a file that was loaded using OPEN.

As an example of its use, you can start EDIT and OPEN your existing AUTOEXEC.BAT file.

1. With **EDIT** running, select the **File** menu and **OPEN**, using either the mouse or the **Alt-F** and **O** keys in succession.
2. The screen appearance for the **OPEN** menu is illustrated on the next page.
3. When you change this filename by re-typing it as ***.BAT** (you can place the cursor just following **TXT** and backspace to erase **TXT**, then type **BAT**), you will see its appearance change to that illustrated below. The file name of AUTOEXEC.BAT now appears in the **Files** list because in this example this is the only BAT file in the C:\root directory.

■ For editing other text, you can use the **Dirs/Drives** list at the right hand side to change to a floppy drive or to a different hard-disk directory. Place the mouse cursor over the black rectangle



The Open item clicked to display the menu. The filename, whose default is *.TXT has been changed to *.BAT so that clicking on OK will show the AUTOEXEC.BAT file in the Files list. You can move to any directory by clicking on the name in the Dirs/Drives list.

at the side of this box (the scroll-bar), click the mouse button and hold it down. Moving the mouse will now drag the box down to see other directory or drive names. Release the mouse button when you see the name you want (which can include **[A]** or **[B]**). Put the cursor on the name and click to make this the current drive or directory.

4. Using keys, press **Tab** to get to the box containing the file name, then press the down-arrow to highlight AUTOEXEC.BAT.
5. When you have the AUTOEXEC.BAT filename in the Files box, double-click with the mouse when the cursor is on the name.

If you are using keys and have highlighted the name, press **ENTER**. You should now see your AUTOEXEC.BAT file.

```
File  Edit  Search  Options                                          Help
                              AUTOEXEC.BAT
@echo off
PATH C:\MSDOS;C:\;C:\WINDOWS;C:\BATS;C:\EDITORS
LH /L:0;1,42400 /S C:\MSDOS\smartdrv
set dircmd=/w/p/o:gn
set temp=c:\temp
LH /L:1,6400 C:\MSDOS\doskey/insert
LH /L:1,56928 C:\MSDOS\mouse
LH /L:1,15904 C:\MSDOS\keyb uk,,C:\msdos\keyboard.sys
LH /L:1,640 keyclick
rem C:\scrngrab\others\camera
LH /L:1,5600 pinch c:\temp
REM pinchf12 c:\temp
REM \scrngrab\hplaser\hpscreen
rem \msdos\catch /p12
REM c:\scrngrab\snapshot\snapshot s=res f=c:\scrngrab\snapshot\snap
C:\MSDOS\SMARTDRV.EXE
cls
dosshell
rem win /s

MS-DOS Editor  <F1=Help> Press ALT to activate menus          N 00001:001
```

The AUTOEXEC.BAT file for a 386 machine as seen on the Edit screen. Note how MEMMAKER (Section 56) has shifted some memory-resident programs into high memory.

Do **not** attempt at this stage to make any changes to this file - the example is intended only to show how you can use **OPEN** to work on an existing file. If you have made absolutely no changes, you can select **File** and **SAVE** to save it back to the disk. If you are in any doubt, simply select **File** and **EXIT**. The AUTOEXEC.BAT file on the disk is not altered when the file is opened, only when a file is **saved** using the name of AUTOEXEC.BAT.

If you are using an editor other than EDIT you will have to find for yourself how to use its OPEN, SAVE and SAVE AS commands.

49 A Simple Batch File

We can look now at how to construct and use a simple batch file. The batch file used as an example will make all the files in a directory called C:\TEXT into read-only form and we can call it, logically enough, RO.BAT. We start, then by running **EDIT**.

With EDIT running, you should be looking at a screen such as was illustrated on Page 174. Do NOT use the **OPEN** command, because you are going to create a new file, not edit an old one. If you have already opened a file, use **New** from the **File** menu to clear this out.

Each command in the batch file needs to use a line of its own, and is typed as it would be if you were typing direct commands - in this case there is only one line to type:

ATTRIB C:\text +R

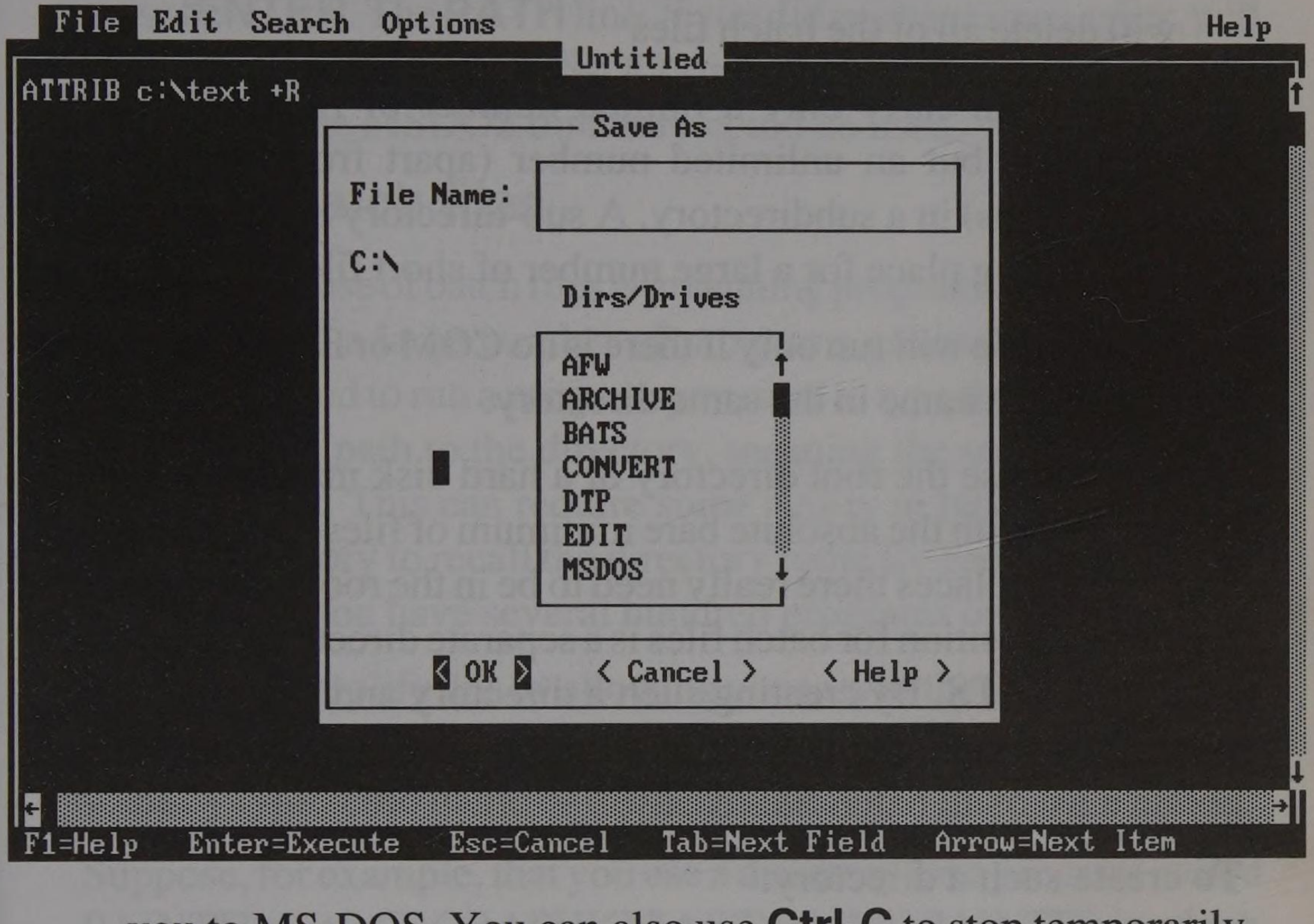
and press **ENTER** so as to take a new line. You can then quit EDIT.

You now need to name the file and save it.

1. Select the **File** menu, and from this use **SAVE AS**.
2. This will bring up a File Name and Directory box, which allows you to select a filename and directory.
3. Double-click on the directory you want to use (in this case, BATS), type the name of the file (in this case RO.BAT) and click on **OK**. You might want to use the C:\root directory to contain your RO.BAT file.
4. Once the file has been saved, it can be invoked by typing **RO** (press **ENTER**), assuming that you are using the directory in which the batch file is recorded, or a suitable path, see later.

■ You can stop a batch file in the middle of its action by pressing the **Ctrl-C** keys (hold down both **Ctrl** and **C**). This brings up a message asking you to confirm that you want to stop the batch file by typing **Y** or **N**. Typing **Y** will stop the batch file and return

The short batch file typed and SAVE AS selected to save this to the disk - a file name will have to be provided, and the directory selected.



you to MS-DOS. You can also use **Ctrl-C** to stop temporarily (perhaps to get a printer ready, for example). When you press the **N** key, the batch file will resume.

- You can also use the **Pause** key, or **Ctrl-S**, to halt a batch file temporarily until you press any other key. There is no option to end the batch file in this case, however.

Using a batch file directory

Batch files are short and they take up very little space on a disk, particularly on a hard disk. It is tempting to keep such files on the root directory (called C:\) of the hard disk, but this is undesirable for several reasons.

- The batch files will be mixed up with other files and with directory names.
- If you use the **DEL** *.* command in error on your hard disk you will delete all of the batch files
- A disk can carry only a limited number of files in its root directory, but an unlimited number (apart from disk space restrictions) in a subdirectory. A sub-directory is therefore the best resting place for a large number of short files.
- A batch file will run only if there is no COM or EXE files of the same main name in the same directory.

You should use the root directory of a hard disk mainly for other directories, with the absolute bare minimum of files. Only the files that SETUP places there really need to be in the root directory. By far the best position for batch files is a separate directory called (for example) **BATS**. By creating such a directory and using it for all batch files you make it easier to work on your batch files and to obtain a list of all batch files easily.

To create such a directory:

1. Type **C:** (press **ENTER**) to ensure that you are using the root directory of the hard disk.
2. Now type **MD BATS** (press **ENTER**). This will create the BATS directory.
3. To go to this directory, type **CD BATS** (press **ENTER**). Any batch file you create while you are using this directory will be placed into this directory.
4. To return to the root directory, type **CD** (press **ENTER**).

In addition, you need to alter the **PATH** line in the AUTOEXEC.BAT file so that the BATS directory is in the search list. A line such as:

PATH C:\;C:\MSDOS;C:\BATS

ensures that directories called MSDOS and BATS will be searched in that order when you type a program name or batch file name and press t**ENTER**. The **PATH** line, if you did not have one earlier, will have been created when MS-DOS 6.0 was installed so as to ensure that files in the MSDOS directory could be used.

Uses of batch files

The simplest use of batch files is in running programs. On a hard disk you will have a large number of programs grouped into different directories, and to run a program you need to type the full filename, including the path to the directory, meaning the set of directories from the root. This can require some agility in typing and a good enough memory to recall the directory route for each program - no mean feat if you have several hundred programs on the disk.

Batch files simplify this by allowing you to type the details just once when you create the batch file. You can then use them whenever you want by typing the name of the batch file.

Suppose, for example, that you use a directory for a program called PAINTIT that is C:\GRAPHICS\ART\PAINT\PAINTIT, in which the PAINTIT program is in a directory called PAINT that is in a directory called ART which in turn is in a directory called GRAPH-ICS. There would normally be other programs and directories in each of the directories as well, but we need only know the path to the program we want.

In this example, typing the full name each time you wanted to use the program would be painfully tedious, and liable to error if you mistyped any letter. By using a batch file called PAINT.BAT and containing this full command line, you ensure that all you ever need to type is **PAINT**, and you could even use the shorter name P.BAT and type **P** to run it if you used nothing else with this letter and could

remember this use. This batch file, PAINT.BAT or P.BAT, can be placed in the BATS directory, so that typing **PAINT** (or **P**) will always run this batch file (but see below for conflicts of names) no matter what directory you start from.

The batch file method of starting a program becomes even more useful if you need to supply any fixed parameters along with the filename. Suppose that when you use the PAINTIT program you need to type the filename and follow it with commands that will indicate the use of a monochrome screen and the use of a directory for files that the program will create (drawing files in this example). A line such as:

C:\GRAPHICS\PAINT\PAINTIT -m C:\DWGS

would mean that the display was monochrome and the files were to be placed in the directory called C:\DWGS. Instead of typing all of this out each time, creating a batch file with this line reduces your effort to typing **PAINT** - and if you named your batch file as P.BAT, you would only have to type **P** and press **ENTER** in order to run it. Using a batch file for each of your main programs makes it far easier to run them that any amount of pointing and clicking with a mouse - and you can determine for yourself what to put into your batch file.

Simply running a program is in itself a useful application of a batch file, and the reduction in effort that it can achieve is very well worth while. You can also greatly reduce the number of errors caused by mistyping a long name, forgetting a long path and so on. Another useful point is that the batch file can deal with the case of requiring a subsidiary program to be loaded.

Suppose, for example, that each time you use a graphics program called DRAWIT you need to run a program that sets up your printer; we'll imagine that the program is called DOTTIT. Starting the DRAWIT program without DOTTIT allows you to make drawings and save files, but does not allow you to print the drawings so that

in order to print you need to leave DRAWIT, run DOTTIT, start DRAWIT again and print the files.

All this is tedious and a strain on the memory, so by creating a batch file called DRAW.BAT you can incorporate these actions automatically, using lines such as:

```
C:\PRINTER\DOTTIT
C:\GRAPHICS\DRAWIT
```

to run the program called DOTTIT which is in the PRINTER directory and then run the program called DRAWIT which is in the GRAPHICS directory.

From now on, you can ensure that these actions are carried out in correct order by simply typing **DRAW** and pressing **ENTER**. This will run the batch file which will execute the two programs in order, and you will never again find yourself having to leave one program, run the other and then start again. The only problem that this system can generate is that if the first program is one that remains in memory, running the batch file for a second time in a session may cause problems (lack of memory, because the memory is filled with identical copies of a program). This can be avoided, but for many purposes the problems do not arise because a program is used only once in a session.

- It is quite possible for a batch file to have the same name as a program file, and for two program files to have the same name but with different extensions. There is no conflict if the files are in different directories - the file that is in the directory that is searched earliest will be run, the others ignored.

- If the files are in the same directory, MS-DOS gives first priority to the COM file. If only EXE and BAT files of the same main name are present, the EXE file will be run. The BAT file is at the end of the chain, hence the usefulness of a separate BATS directory which is searched early in the PATH statement.

50 Batch File Parameters

A **parameter**, as far as running programs is concerned, is a piece of information that is added to the program name. The (imaginary) example of **PAINTIT -m C:\DWGS** makes use of the two parameters **m** and **C:\DWGS** to configure the program for a monochrome screen display and for placing data files in the C:\DWGS directory. This form of use of parameters is quite common, and for this type of example, the parameters are fixed - you would normally retain the same display and always want to put the files into the same directory.

Some programs, however, require parameters that might not remain unchanged. A word-processor might require you to specify the directory for its data files, and you would probably need to keep more than one directory, such as one for letters, one for financial documents, one for addresses, one for short articles and so on.

- There would be little point in entering a fixed name into the batch file line that started such a program; what you need is a method of specifying the parameter as you type the batch file name, just as you would specify it when you typed the program name.

Batch files cater for this by using the parameter names **%1** to **%9** (**%0** is reserved to carry the name of the batch file itself). Suppose, for example, that you have a word processor called SHRDLU and you normally start it by typing its name followed by a space and then the name of the directory that contains the files you want to use with it. With just one parameter to use, the main line of this batch file will be something like:

C:\WORDS\SHRDLU %1

in which **%1** is a placeholder for the name that you will type when you start the batch file running. If you call this batch file PEN and you start it by typing **PEN LETTERS**, then the name LETTERS is the first parameter name, and it is assigned in place of %1, so that the effect is as if you had typed:

C:\WORDS\SHRDLU LETTERS

starting the word processor and making use of the correct data directory.

This provides very much more flexibility than the use of a fixed name within the batch file, and the parameter %1 can be used more than once in the batch file if this is required. You might, for example, want to copy all of the files in whatever directory was selected to a floppy after ending use of the word-processor, in which case the lines in the batch file:

```
C:\WORDS\SHRDLU %1
COPY %1\*.TXT A:
```

would see to it that the directory was chosen and later used to copy all files with the extension of TXT (in that directory) to a floppy in the A: drive.

- You need to remember what the parameters stand for. In the example, %1 will be a directory name such as LETTERS, and in a filename it will be followed by the backslash (\) and then by the name of the file itself. When the batch file substitutes LETTERS for %1, the complete filename must be in its correct form.
- Note that when you use a batch file along with DOSSHELL (see Part 4) you can ensure that you are prompted for each parameter, with a reminder of what is needed.

Whether omitting a parameter will cause an error message or not depends on the command that uses the parameter. For example, if the command is **DIR %1**, then omitting a parameter when you call the batch file will provide a directory listing for whatever directory you happen to be in, rather than an error message. You will get an error message only if the absence of a parameter makes the command impossible to execute.

50 Batch File Parameters

This is not necessarily obvious. For example, if you used **DEL B:\%1**, then omitting a parameter will be taken as meaning *.*, all files to be deleted, and you will get the usual message about whether you want to do this or not. This could lead to unwanted and unwitting deletion of all files in a directory - and you might be using the root directory of the hard disk at the time. More advanced batch file programs can detect the absence of parameters and abandon the batch action with a warning message.

- This book does not cover advanced batch file work. There are several commands that can be used only in batch files, but their use demands some experience of programming, see later.

The number of parameters that you can use in a straightforward way is 9, %1 to %9 inclusive. You can use more parameters by recycling numbers with the **SHIFT** command. This is most unusual because very few programs require so many parameters that would need to be changed each time you used the batch file - it is more usual for all but one or two parameters to remain fixed.

- Note that a name that you supply for a parameter must **not** contain any spaces. A space is **always** taken as a separator between parameters. This can cause problems if you want to write a batch file that uses parameters other than filenames.

Using ECHO 51

Batch files can make use of a set of commands of their own, many of which can be used only in batch files. One useful command is **ECHO** which exists in several forms.

@ECHO OFF will suppress the appearance of the batch file commands on the screen. Whether you use this or not is a matter of choice. Some batch file users like to see the commands appear, others do not. It is an advantage to leave the command out when you first try out a batch file, because the appearance of the commands on the screen indicate the progress of the file. Once you have tested the batch file thoroughly you can use **@ECHO OFF** to suppress the printing of these command lines.

ECHO has another use, to make text appear on the screen even when you have used **@ECHO OFF** as a first line. The text in this case is not the command lines, which remain suppressed, but any text you like to put in. For example, if your batch file contains the line:

ECHO Do you want to copy this file to A:\ ? (Y/N)

then the words of the message (but not the word **ECHO**) will appear when this line runs. You can use this to deliver reminders (a simple HELP system), warnings (about deleting files, for example) or error messages (about absence of parameters, for example) or any other use you can think of. You can make a batch file that consists only of **ECHO** lines in order to provide outline documentation for a program. You can call such a batch file from another batch file if required, using **CALL** followed by the name of the new batch file.

Batch files are capable of very much more than has been illustrated here, but only if you have some experience of programming in a language such as BASIC. Since this book is aimed at the less experienced, advanced batch files will not be covered here, but the *Pocket Book of MS-DOS 6.0* provides an outline course in batch programming methods which will be useful if you decide to try your hand after gaining experience of using MS-DOS 6.0.

- 6 -

Memory

52 PC Memory

The design of the original PC machines allowed for a total of 1 Mbyte of memory, of which 640 Kbyte could be RAM used for programs. The remainder is used for ROM, permanent memory, storing parts of the operating system so that the machine can be booted and load in MS-DOS. To ensure compatibility of programs, later machines have had to accept the limitation of 640 Kbyte for program use despite the fact that the chips now being used provide for much more memory to be installed.

The problem was initially caused by placing the ROM at the top of the 1 Mbyte of memory (instead of at the start), but is now due as much to MS-DOS as to the design of the machines, and is carried over into MS-DOS 6.0.

A program that is written to run under MS-DOS control will need only this standard amount of memory, 640 Kbyte of read/write (RAM) and 384 Kbyte of ROM. Additional memory is needed only for :

1. The use of MicroSoft Windows versions 3.1 onwards
2. The use of memory as a cache, temporarily storing data whose permanent place is on the hard disk
3. The more advanced operating systems such as IBM OS/2, Windows 4.0, MS-DOS 7.0, and Microsoft NT which will eventually replace the earlier varieties.

- If your requirements are for MS-DOS programs only, with no need to speed up disk read and write actions with a cache, you do not really need MS-DOS 6.0 nor any added memory.

Modern machines that use the 80386, 80486 or Pentium chips are likely to be provided with and used with Windows 3.1 (or NT) and other later operating systems. They need additional memory and MS-DOS 6.0.

The additional memory for such machines is called **extended** memory.

- Do not be confused by an older system called **expanded** memory that could be used with less-capable machines.

The modern form of extended memory is fitted directly on the main board of the computer (the motherboard) and comes in units called **SIMM** (single inline memory module). Each SIMM can add either 1 Mbyte or 4 Mbyte, depending on size. On most machines you must add SIMM memory in pairs or in blocks of four, so that memory sizes can be 2, 4, 8, or 16 Mbyte, not intermediate values.

A modern PC must still remain compatible with the older designs, and this is why the PC design has prevailed since its introduction in 1981. The need for compatibility, however, makes it more difficult to use added memory, because of the way that the first 640 Kbyte of RAM is separated from any additional RAM.

There is no provision built into MS-DOS for controlling extended memory, and this capability has to be added by a program called a memory manager. The main memory manager of MS-DOS 6.0 is called **HIMEM.SYS**, and when you install MS-DOS 6.0 this manager will also be installed.

- From this point, we shall assume that you are using a machine (or intend to) that has been fitted with extended memory. If you intend to use Windows, you will need a total memory of 4 Mbyte or more, and for the more advanced operating systems you will probably need at least 8 Mbyte.

- A few programs can still use the older type of expanded memory. You can also gain some advantage by using a small amount of memory of this type (called an Upper Memory Block or **UMB**) for some programs. MS-DOS 6.0 allows you to use extended memory for these actions.

53 Memory Reports

MS-DOS 6.0 contains, among its utilities, a program called **MEM**, which you can use to analyse the way that your computer handles memory. If you use **MEM** in its simplest form, you will get the information typically illustrated below for a 386 machine. This report was taken after MS-DOS 6.0 was installed but before doing anything to make better use of memory.

```
C:\>mem

Memory Type          Total =  Used   +  Free
----------------   ------   ------   ------
Conventional          640K      54K     586K
Upper                  28K       7K      22K
Adapter RAM/ROM       356K     356K       0K
Extended (XMS)       4096K     496K    3600K
----------------   ------   ------   ------
Total memory         5120K     912K    4208K

Total under 1 MB      668K      60K     608K

EMS is active.
Largest executable program size          586K  (600368 bytes)
Largest free upper memory block           22K   (22208 bytes)
MS-DOS is resident in the high memory area.

C:\>
```

A typical MEM report from a 386 machine which has a total of 5 Mbyte installed

This is a report on the main (system) memory, showing that the conventional memory is 655360 bytes (640 Kbyte), of which some 586 Kbyte is available for programs, despite the use of some memory-resident programs (such as the PINCH program that was used to capture the screen display).

- In this example, there is a large amount of conventional memory available because much of MS-DOS 6.0 has been relocated to the first 64 Kbyte of extended memory, the region called **Upper** in this report.

- This Upper memory comes about by chance, because the early models of PC/AT machines were able to use this piece of memory without the need for an elaborate manager system. The feature has been retained, and though this memory is a part of the main extended memory it is always regarded as being different because it can be more easily used by programs (DOS programs) which would normally be able to use only the first 640 Kbyte of memory.

- The remainder of the extended memory can be used only by programs that have been specially written to use it. These programs are not compatible with older MS-DOS programs, and most of them are designed to use Windows 3.1 to manage their memory requirements.

1. The machine is fitted with a total of 5 Mbyte of RAM memory, and the report from **MEM** shows 4096 Kbyte (4 Mbyte) of extended memory of which only 496 Kbyte is reported as being in use.

2. The figure shown as **Total Under 1 Mbyte** consists of the normal 640 Kbyte of conventional memory plus the 28 Kbyte of upper memory that remains available.

3. The report shows that **EMS is Active**, meaning that expanded memory can be simulated if needed because the necessary memory manager is running.

Though you can use **MEM** to provide much more detailed reports on the memory of a machine and how it is utilised, there is little point in obtaining these reports unless you know enough about memory to be able to use them. Since this book is aimed at the less-experienced, we shall concentrate on how MS-DOS 6.0 can be used to make memory available, rather than on how an expert can squeeze a little more memory out of a system by interpreting these reports.

53 Memory Reports

Memory managers

The use of one or more memory manager programs is essential if there is to be **any** use of memory above the 640 Kbyte barrier. The memory managers that you will need for 386 (and above) machines are supplied along with MS-DOS

MS-DOS 6.0 is provided with two memory managers. HIMEM.SYS is an extended memory manager that allows access to extended memory for DOS and for Windows.

- Remember that making memory available for Windows means that memory is made available for any program that is designed to make use of Windows, and all of the leading software is either now available in this form or is in the process of being converted.

HIMEM.SYS also manages extended memory so as to avoid conflicts between programs, preventing programs from trying to make use of the same part of memory. HIMEM.SYS must be run (in the CONFIG.SYS file) **before** there is any attempt to make use of extended memory on any 80286, 80386 or 80486 machine.

This means that the first, or a very early, line of your CONFIG.SYS file should read:

DEVICE=C:\MSDOS\HIMEM.SYS

assuming that the HIMEM.SYS file exists in the C:\MSDOS directory of the hard disk.

- You can save yourself much trouble simply by ensuring that this is the first line of the CONFIG.SYS file, and checking that it is still the first line after new programs have been installed, because SETUP programs often alter your CONFIG.SYS file.
- There are still some programs about that modify CONFIG.SYS without notifying you and though it is unlikely that they would

delete your own CONFIG.SYS they might substitute another with the HIMEM.SYS omitted, or with your own CONFIG.SYS instructions tacked on at the end. It is always advisable to keep several backups of your CONFIG.SYS file on write-protected disks.

The installation of MS-DOS 6.0 ought to have attended to all this, but it is wise to check. If you want to see your CONFIG.SYS file without necessarily editing it, use the command:

TYPE C:\CONFIG.SYS

assuming that CONFIG.SYS is in the root directory of the hard disk, as it ought to be.

The straightforward and simple **HIMEM.SYS** line illustrated above is as much as most users ever need. If problems arise that do not seem to arise from conflicts of memory-resident software (see later) you may need to make use of some of the optional additions to the HIMEM command.

- At one time you had to edit these in for yourself, but MS-DOS 6.0 has a utility called MEMMAKER that will modify the HIMEM.SYS line and set up your memory for optimum use.

The only possible reasons for editing the HIMEM.SYS line arise out of special needs, which might be:

1. Fine-tuning for faster use
2. Problems with older software
3. Problems with some hardware.

Sorting out these requirements is definitely **not** a task for the inexperienced, which is why the MEMMAKER utility is so welcome

If you need to know what modifications can be made using HIMEM.SYS and for what purposes, a full description is contained in the *Pocket Book of MS-DOS 6.0*.

54 Problems with HIMEM.SYS

Problems with HIMEM.SYS are almost all concerned with conflicts of memory use, particularly if another memory manager (for an expanded-memory board) is present. These conflicts are less common now that writers of software realise the need to work to agreed standards.

- No machine in the 386/486 class should require to use an expanded memory board. Resist the temptation to use an expanded memory board just because you happen to have one from an old machine, or you have bought one at a low price.

The SETUP program for MS-DOS 6.0 will not even install HIMEM.SYS if another extended memory manager program is active. If you have just upgraded to MS-DOS 6.0 and found that HIMEM.SYS is not present you will have to copy it by using EXPAND.EXE. Find the MS-DOS 6.0 disk that contains the file called **HIMEM.SY_** file, and use **EXPAND** on this file to copy HIMEM.SYS to the hard disk.

1. Place into the A: drive the floppy disk that contains the file called HIMEM.SY_
2. Type **A:** to change to the A drive
3. Type the command:

   EXPAND HIMEM.SY_ C:\MSDOS\HIMEM.SYS

4. Wait for the file to be expanded and copied.

One common source of difficulty is a conflict with some memory-resident program that is loaded either in the CONFIG.SYS or AUTOEXEC.BAT files, and a useful way of checking is as follows:

1. Reboot and hold down the **F8** key when the **Loading MSDOS** message appears. This will allow you to determine which CONFIG.SYS lines you want to use.

2. As each line of CONFIG.SYS appears on the screen, you will be asked whether you want to use it or not (press **Y** or **N** keys).
3. To start with, press **Y** for each line in CONFIG.SYS, but **N** for AUTOEXEC.BAT.
4. Check to find if the conflict problem still exists. If it is no longer present, it has been caused by a line in the AUTOEXEC.BAT file which loads in a memory-resident program
5. If the problem persists at step 4, it must be due to a line in CONFIG.SYS.

Sorting out memory conflict problems is never simple, but once you know which file is at the root of the problem the procedure is one of monotonous routine rather than inspiration.

The method is to make an alteration, reboot and then look for the problem. Each time you reboot, you need to change one line of the file that you know contains the line that causes the problem (AUTOEXEC.BAT or CONFIG.SYS).

1. Find the lines that load programs or drivers. In AUTOEXEC.BAT these lines will either start with **LOADHIGH** (or **LH**), or consist of the name of a program. In CONFIG.SYS look for lines that start with **DEVICE** or **DEVICEHIGH**.
2. Working from the bottom upwards, one line at a time, edit the file. Place the word **REM** ahead of a command that loads a program or driver
3. With one line changed (**REM** as the first word), reboot and look for the problem.
4. If you find that the problem still exists, edit again. Remove the **REM** from its line and type it into the next higher line that loads a program or driver.

5. At some stage you will find that adding **REM** to a line causes the problem to disappear when the machine is booted up. This is the troublesome line.

Once the problem has been isolated you can deal with it.

- You might be able to dispense with the line - it might load a utility you do not need.
- Another way of dealing with it is to relocate the line. Surprisingly often, the order of lines in CONFIG.SYS or AUTOEXEC.BAT is important. A line that may be troublesome when placed early in its file can often present no problems if it is moved to a position near the end of the file.
- Some care is needed. Never place **REM** ahead of the line that contains **HIMEM.SYS**, and never place any line before the **HIMEM.SYS** line.
- Remember also that you have to reboot to see any change in CONFIG.SYS take effect, and it is also desirable to reboot to see the new AUTOEXEC.BAT take effect, though you can type AUTOEXEC as a direct command.

If you still encounter problems, check the READ.ME files of MS-DOS 6.0 to see if there is a known problem with your make of computer or the ROM chip (the BIOS chip) that it uses. The READ.ME file contains many examples of known incompatibilities and ways of getting around them.

EMM386 is the other MS-DOS 6.0 memory manager utility. It is intended for use by 386 and 486 machines (or higher) only; it **cannot** be run on 8088, 8086 or 80286 machines. Any attempt to run EMM386 on such machines will cause an error message.

The purpose of EMM386 is to use some of the extended memory of a 386 or 486 machine for two distinct purposes:

- To make use of the Upper Memory Block (UMB). This is a way of using some extended memory so that it appears to be within the first Mbyte.
- To use of extended memory (if it is available) as expanded memory for the programs that can use expanded memory, such as DOS versions of AutoSketch, DeLuxe Paint-2, etc. Windows versions of programs will use extended memory that is managed by Windows itself.

The use of UMB memory is valuable, because you can load and run several small programs into that region. Programs do not have to be specially written to be able to run in UMB memory, though there are a few that will not. If you arrange to move short memory-resident programs and utilities to UMB, conventional memory can be released, allowing large programs with no extended or expanded capabilities to make full use of the conventional memory.

- EMM386 **must not** be used along with any other expanded memory managers. A machine using the 386 or 486 (or later) chip should never be fitted with an expanded memory board.

Remember that you cannot use the same memory for two different purposes. If you install EMM386 it will itself take around 80 Kbyte of extended memory that is not free to be used for anything else, and the extended memory that it converts to UMB or expanded cannot be used also as extended memory in its own right.

- You can, however, use EMM386 for UMB uses only, leaving the rest of the extended memory free. This is well suited to machines with only 2 Mbyte of RAM. If you have 3 Mbyte or more of RAM, full use of EMM386 along with HIMEM can provide very flexible options for large DOS programs that need expanded memory. Remember, however, that such programs are a dying breed - newer programs are being revised to run with Windows.

Windows is always a special case, and EMM386 should **not** be used for expanded memory if you are intending to use only Windows programs. You can use EMM386 to advantage, however, to provide UMB memory for Windows.

EMM386 is installed by a line in the CONFIG.SYS file that reads:

DEVICE=EMM386.EXE /NOEMS

This must follow the **HIMEM** line in the CONFIG.SYS file because EMM386 cannot take effect unless HIMEM is managing extended memory.

This will start up the EMM386 manager for Upper Memory Block management only, with no expanded memory supplied. This is the way that most PC users will make use of EMM386. The alternative, if you **must** have expanded memory, is to use the line:

DEVICE=EMM386.EXE /RAM

- This must be used if you are running Lotus 1-2-3 V.3.1, since this program requires some expanded memory.

If you want to use UMB memory for any purpose, you need to notify this also in another part of the CONFIG.SYS file. You need either:

1. To add **DOS=UMB** to the CONFIG.SYS file OR
2 To change an existing **DOS=HIGH** to **DOS=HIGH,UMB**

This step connects the use of DOS in the conventional memory to its use in the UMB and is **essential** if you are to make any use of the UMB.

For most users, the default values that EMM386 uses are sufficient, and the only decision that needs to be made is whether or not to use the **/NOEMS** option. The options and additions that can be used with EMM386 are useful only if they are required and if you know how to use them. Programming experience is needed to make effective use of some options.

DEVICEHIGH and LOADHIGH

These are optional forms of the **DEVICE=** form of command for the CONFIG.SYS file, and a way of loading memory resident (TSR) programs in the AUTOEXEC.BAT file, respectively. The simple forms are:

DEVICEHIGH=C:\MSDOS\SETVER.EXE in CONFIG.SYS

LH KEYCLICK in AUTOEXEC.BAT

- note that **LOADHIGH** can be abbreviated to **LH.**

- **DEVICEHIGH** will place a driver into upper memory if such memory is available - if no memory is available the action is exactly the same as **DEVICE**, loading the driver into conventional memory.
- The **LOADHIGH** (**LH**) command performs a similar action for AUTOEXEC.BAT files.

56 Using MEMMAKER

MEMMAKER is a program that automates the use of HIMEM.SYS and EMM386. You can use MEMMAKER only on a 386, 486, or higher, machine. If you attempt to use MEMMAKER on a 286 or lower machine you will be met with a polite refusal.

- The use of MEMMAKER will always be advantageous unless you are sufficiently expert in the use of HIMEM.SYS and EMM386 to be able to use the options and know which ones to use.
- These options present very great difficulties for most PC users so that MEMMAKER has been devised to ensure that the less skilled user can set up a machine to its optimum capabilities.

Because MEMMAKER is concerned with organising the use of memory, it must be used with the memory as clear as possible. This means that the normal memory-resident programs should be installed but you **must not** be using DOSSHELL or Windows. If you normally use a Network, it should be installed.

Once you have the machine running with the normal memory-resident programs, run MEMMAKER as follows:

1. With the **C:\>** prompt showing, type **MEMMAKER** and press **ENTER**.
2. You will see the opening screen. This summarises the MEMMAKER action.

- Note in particular that if you add, later, any software that will become memory-resident (including **DEVICE** lines in the CONFIG.SYS file) you should run **MEMMAKER** again.

3. You are given the choice of **Express Setup** or **Custom Setup**. Custom Setup is used only if you are knowledgeable about the way that you want memory to be used, and for the majority of PC owners Express Setup is preferable.

```
Microsoft MemMaker

Welcome to MemMaker.

MemMaker optimizes your system's memory by moving memory-resident
programs and device drivers into the upper memory area. This
frees conventional memory for use by applications.

After you run MemMaker, your computer's memory will remain
optimized until you add or remove memory-resident programs or
device drivers. For an optimum memory configuration, run MemMaker
again after making any such changes.

MemMaker displays options as highlighted text. (For example, you
can change the "Continue" option below.) To cycle through the
available options, press SPACEBAR. When MemMaker displays the
option you want, press ENTER.

For help while you are running MemMaker, press F1.

                Continue or Exit? Continue

ENTER=Accept Selection  SPACEBAR=Change Selection  F1=Help  F3=Exit
```

The Opening Screen of MEMMAKER with its notes.

4. You will be asked if you use any programs that require expanded memory. The default answer is **Yes**, but change this to **No** if you normally use Windows programs, or if you make little use of programs that require expanded memory.

■ For example, AutoSketch 3.0 uses expanded memory, but only for large drawings that need the additional memory. If your drawings are small and/or simple, you can use AutoSketch without expanded memory.

5. MEMMAKER will check for the presence of Windows, and make allowance for this.

6. You are then notified that pressing **ENTER** will re-start the machine with the improvements implemented - there is a delay between pressing **ENTER** and seeing this happen.

Your computer should re-start normally, and you should see the MEMMAKER screen appear again to tell you that the settings will be used from then on.

- When you leave MEMMAKER your CONFIG.SYS and AUTOEXEC.BAT files will have been altered to place as many memory-resident programs as possible in the higher memory addresses (High memory and UMB memory).
- In addition, some options may have been added to the HIMEM.SYS and/or EMM386 lines.

If the computer does not restart normally, perhaps hanging up or giving unusual messages during booting up, reset again. When the machine starts again you will see the MEMMAKER screen re-appear, asking you to opt for more conservative settings.

- NOTE that MEMMAKER will re-start even if the machine has been switched off and then on again.

In the course of using MEMMAKER, the MEMMAKER screen will appear several times until all of the settings have been made and you confirm that the machine is running normally. Old CONFIG.SYS and AUTOEXEC.BAT files are preserved as CONFIG.UMB and AUTOEXEC.UMB file names **in the MSDOS directory**.

MEMMAKER will normally make a considerable number of changes to both the CONFIG.SYS and the AUTOEXEC.BAT files. Some of these may be to add options to the HIMEM.SYS or EMM386 lines, others are to place memory-resident programs in high memory or UMB space.

Using MEMMAKER 56

If, after using MEMMAKER, you encounter trouble, you can use MEMMAKER to try again with what it calls less aggressive or more conservative settings. This makes less use of upper memory to avoid conflicts. You lose a little memory in this way, but you will also lose the conflicts. Only a knowledgeable and experienced PC user is likely to improve on the MEMMAKER settings, and only by a few Kbyte.

57 Hard Disk Utilities

Modern hard disks for PC machines are mainly of a type called **IDE** (integrated device electronics) which are much easier to install than the older varieties called MFM or RLL. Installing a new hard disk is not a task for the inexperienced, and since you could not have installed MS-DOS 6.0 unless you had a hard drive installed and formatted the processes will not be described here.

- If you want to change or add a new hard disk, the methods are outlined in the *Pocket Book of MS-DOS 6.0*.

- The emphasis in this and the following sections will be on the **use** of the hard disk. MS-DOS 6.0 offers several new utilities that are designed to make this much easier.

MS-DOS 6.0 contains new **MSBackup** utility that is a very greatly improved version of the old BACKUP and RESTORE.

- There is still a RESTORE command that will read disks made by some older versions of BACKUP. These were included in earlier versions of MS-DOS.

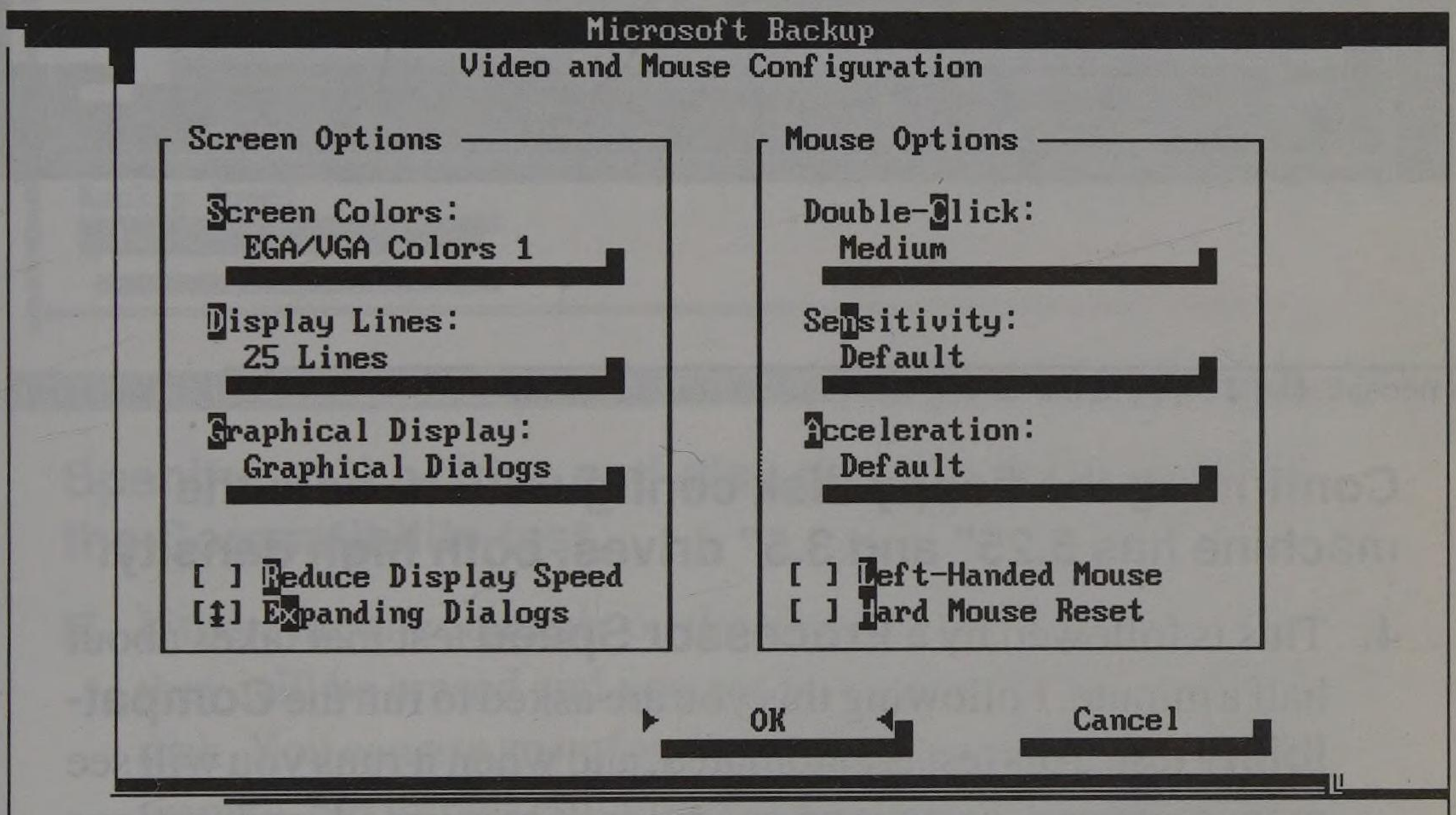
If you have just installed MS-DOS 6.0 into your computer, you should as soon as possible configure the MSBACKUP command, even if you do not intend to use it right away. The configuration process sets up and tests the command so that you can be certain that if a full backup is needed it will be successful.

- Backup exists in two forms, the DOS form (MSBACKUP) and the Windows form (MWBACKUP). You can choose which to install (or to install both) when you install MS-DOS 6.0. This book deals with the MS-DOS version only.

Before configuring MSBACKUP, if you have been using DOSSHELL or Windows, exit from it and continue as follows when the **C:\>** prompt appears:

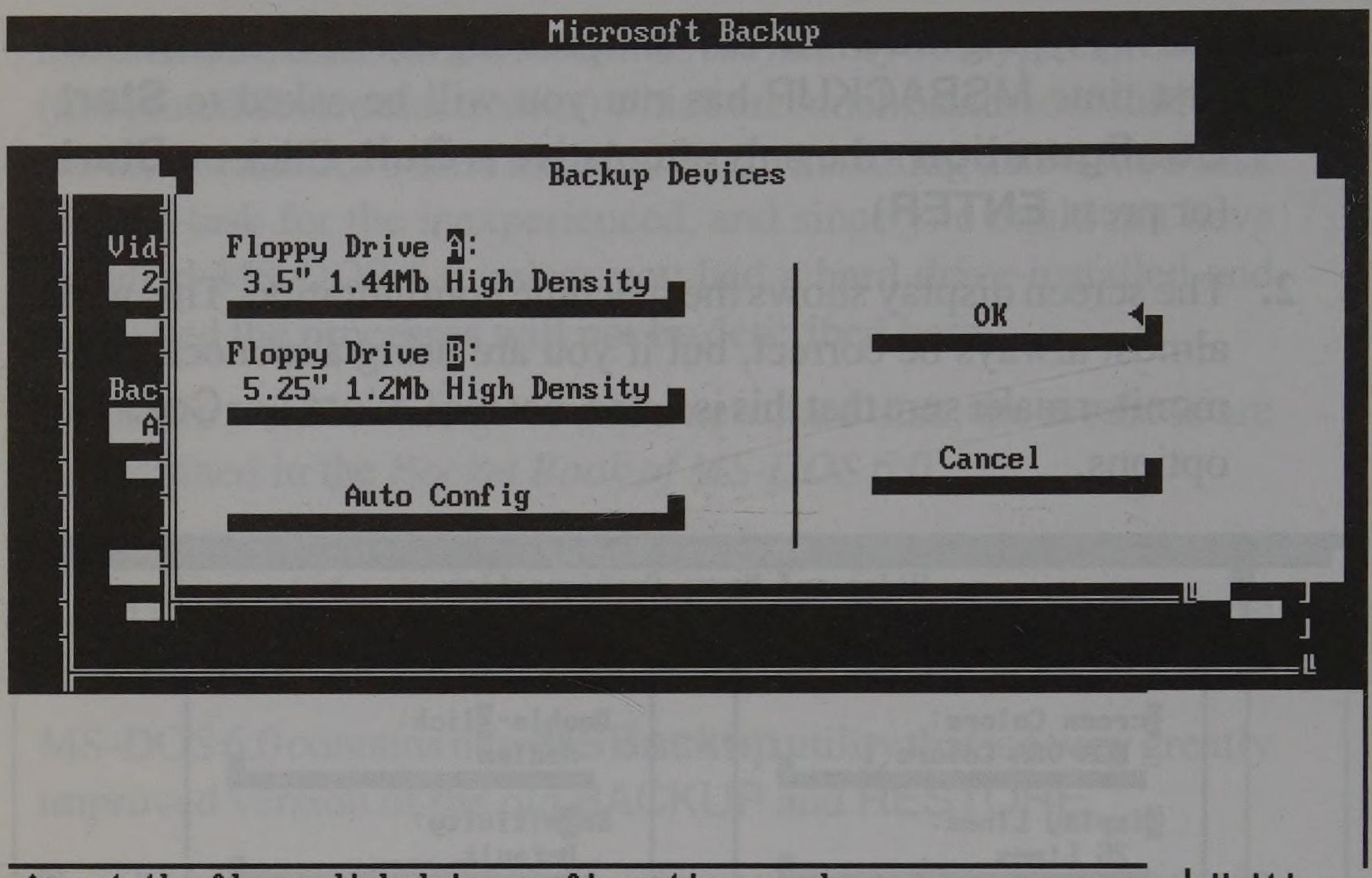
1. Start by typing **MSBACKUP** and pressing **ENTER**. If this is the first time MSBACKUP has run you will be asked to **Start Configuration** - the only alternative is **Quit**. Click on **Start** (or press **ENTER**).

2. The screen display shows the machine configuration. This will almost always be correct, but if you are using a monochrome monitor make sure that this is listed, not the Colors1 or Colors2 options.



The Machine configuration display of MSBACKUP, which gives an analysis of the way that your computer is set up.

3. You are then asked to remove all floppy disks from the drive(s) so that Backup can determine how to detect when a disk has been changed. When this has been done you are asked to confirm the floppy disk configuration

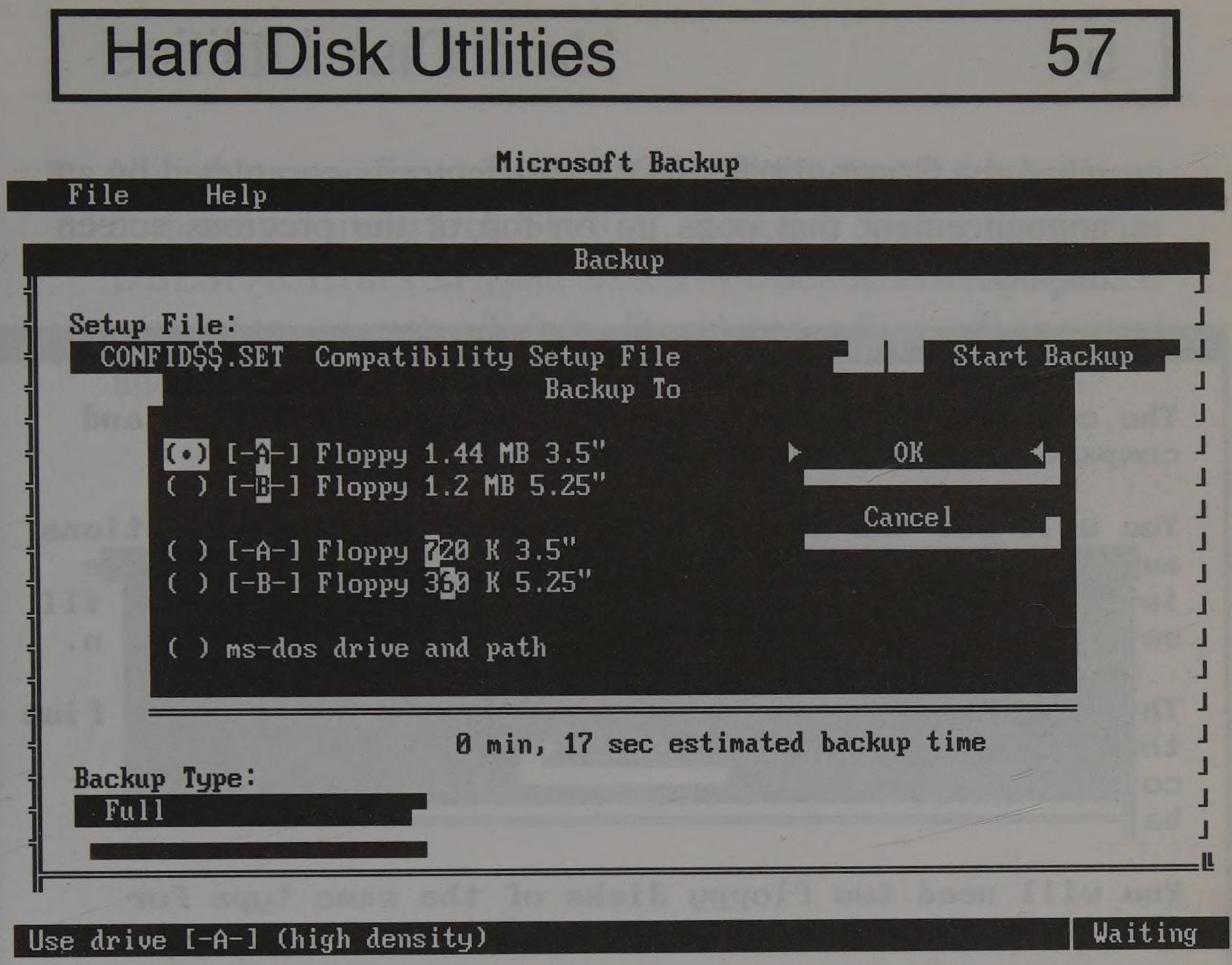


Confirming the floppy disk configuration -here the machine has 5.25" and 3.5" drives, both high density.

4. This is followed by a **Processor Speed** test that takes about half a minute. Following this you are asked to run the **Compatibility** test. This test is automated, and when it runs you will see menus appear, selections made and confirmed and actions started **without any need for you to do anything**.

- Your only action is to feed in floppy disks. For the test, two 3.5" 1.4 Mbyte disks or their equivalent will be needed.

5. Click with the mouse cursor on **Start Test** (or press **ENTER** when this option is highlighted). When you are asked, select the floppy drive you want to use, and the disk density. After a short time you will be asked to insert a disk and the backup process will start.

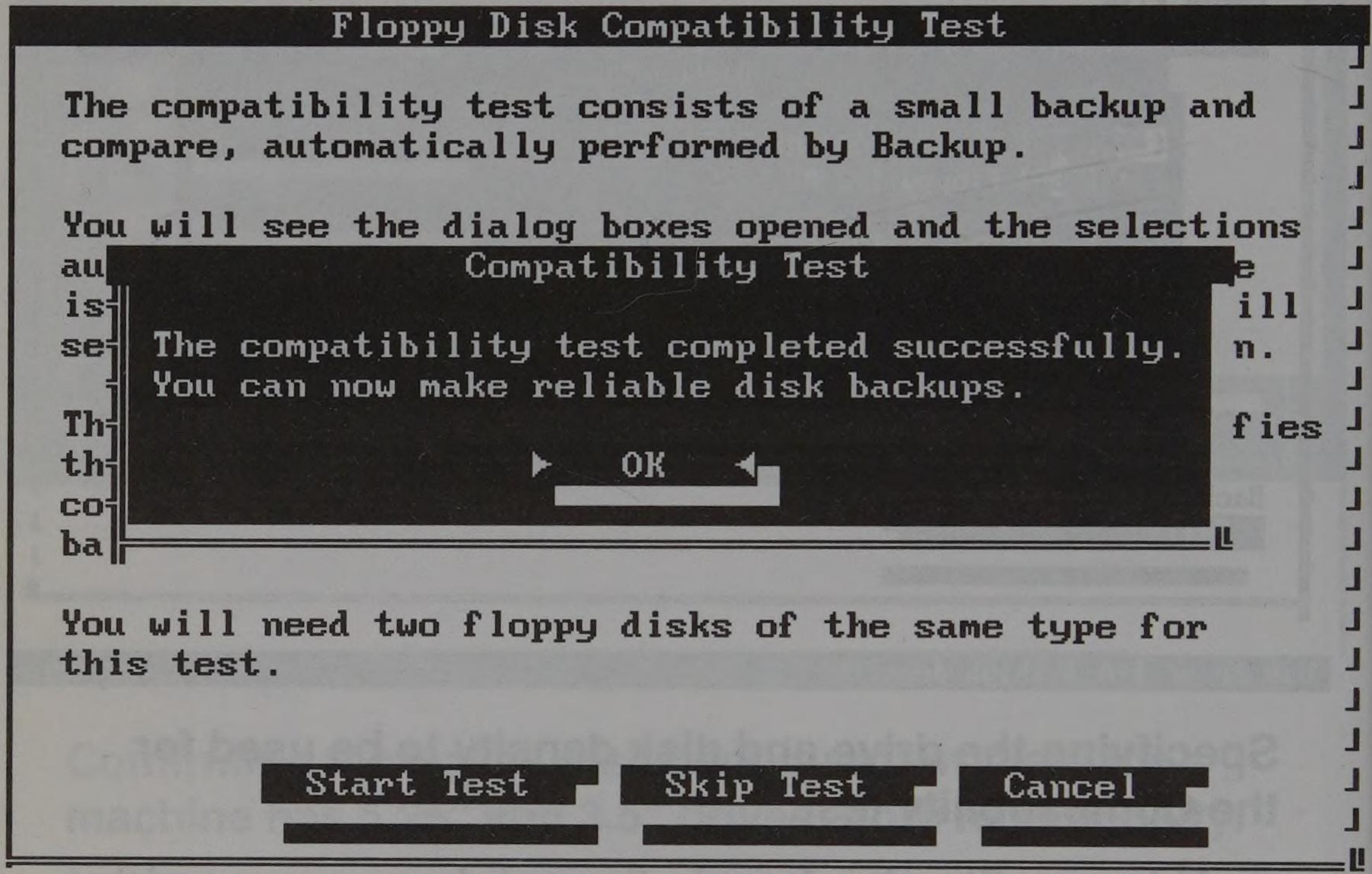


Specifying the drive and disk density to be used for the Compatibility test.

- If there are files already on the floppy disk, you are warned that they will be erased and you are given the chance to replace the disk. You can use an unformatted floppy as a backup disk if you want to, but this takes longer because the disk will be formatted as the backup is made.

6. The screen display will show the progress of the test backup. You will be asked to change disks when the first disk is full - the drive light remains on during this time unless you wait too long. When the backup is over you have to click to confirm, and the recovery process then starts.
7. You will be asked to put the first disk in the drive again and click on **Continue**. This starts the **Restore** action and you will be asked to change disks at the appropriate time. You are informed

when the Compatibility test is satisfactorily completed by an announcement that pops up on top of the previous screen display.


Floppy Disk Compatibility Test
The compatibility test consists of a small backup and compare, automatically performed by Backup.
You will see the dialog boxes opened and the selections
Compatibility Test
The compatibility test completed successfully.
You can now make reliable disk backups.
OK
You will need two floppy disks of the same type for this test.
Start Test
Skip Test
Cancel

The end of test announcement which pops over the main menu of MSBACKUP.

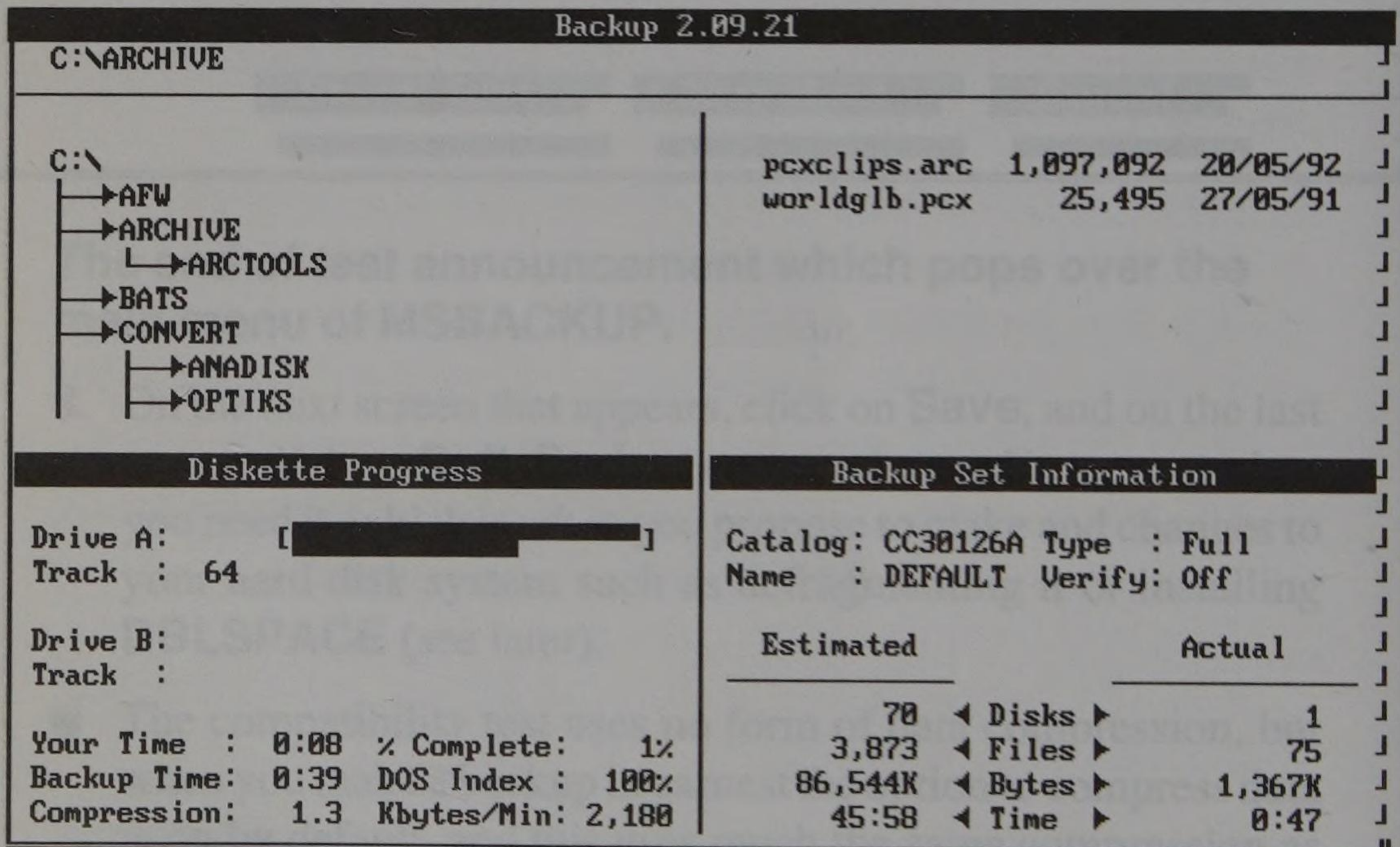
8. On the next screen that appears, click on **Save**, and on the last screen click on **Quit**. Backup can now be used in earnest when you need it, which is when you propose to make and changes to your hard disk system such as defragmenting it or installing **DBLSPACE** (see later).

■ The compatibility test uses no form of data compression, but when you make a backup in earnest the option to compress data is on by default, and this uses much the same compression as Doublespace - 1.9 Mbyte of files will fit easily on a 1.2 Mbyte floppy, for example.

- Note that you **cannot** use for MSBackup a floppy that has been compressed using DBLSPACE, see later. Such a floppy has DBLSPACE files on it and is seen by Backup as a used disk. If you opt to use such a disk the Doublespace files will be wiped and the disk will be treated as a normal disk.

To use **MSBACKUP** for a total backup you must start the program from MS-DOS. If you have been using DOSSHELL or WINDOWS you must exit back to the MS-DOS **C:/>** prompt. Prepare a large number of disks and label them with sequential numbers. The procedure for a full backup is:

1. On the initial screen, select the hard drive you want to backup, usually **C:**.
2. Double-click to see the full file list. Select the floppy drive and disk type.
3. The top right box that contains the phrase **start backup** will read **Start Backup** when you are ready to start. Click on this.
4. You will be asked to insert the first floppy, and warned if it contains any data. The display will show the estimated number of disks required - 70 in this example.



The *number of disks* figure in this display shows why so many user prefer a tape or floptical backup system.

- If you do not have this number of floppies you can opt to cancel the backup.

5. Change disks as prompted until the last disk has been used. The estimated number may change as the backup progresses (often to a smaller number than was originally estimated).

Use the **Quit** option when the **MSBACKUP** screen returns, and place the backup disks in a safe place.

Partial Backup

A full backup is a time-consuming and floppy-consuming process unless you are working with 21 Mbyte (or larger) Floptical disks with the **compression** option of Backup in use. Backup, however, allows you a large range of choices:

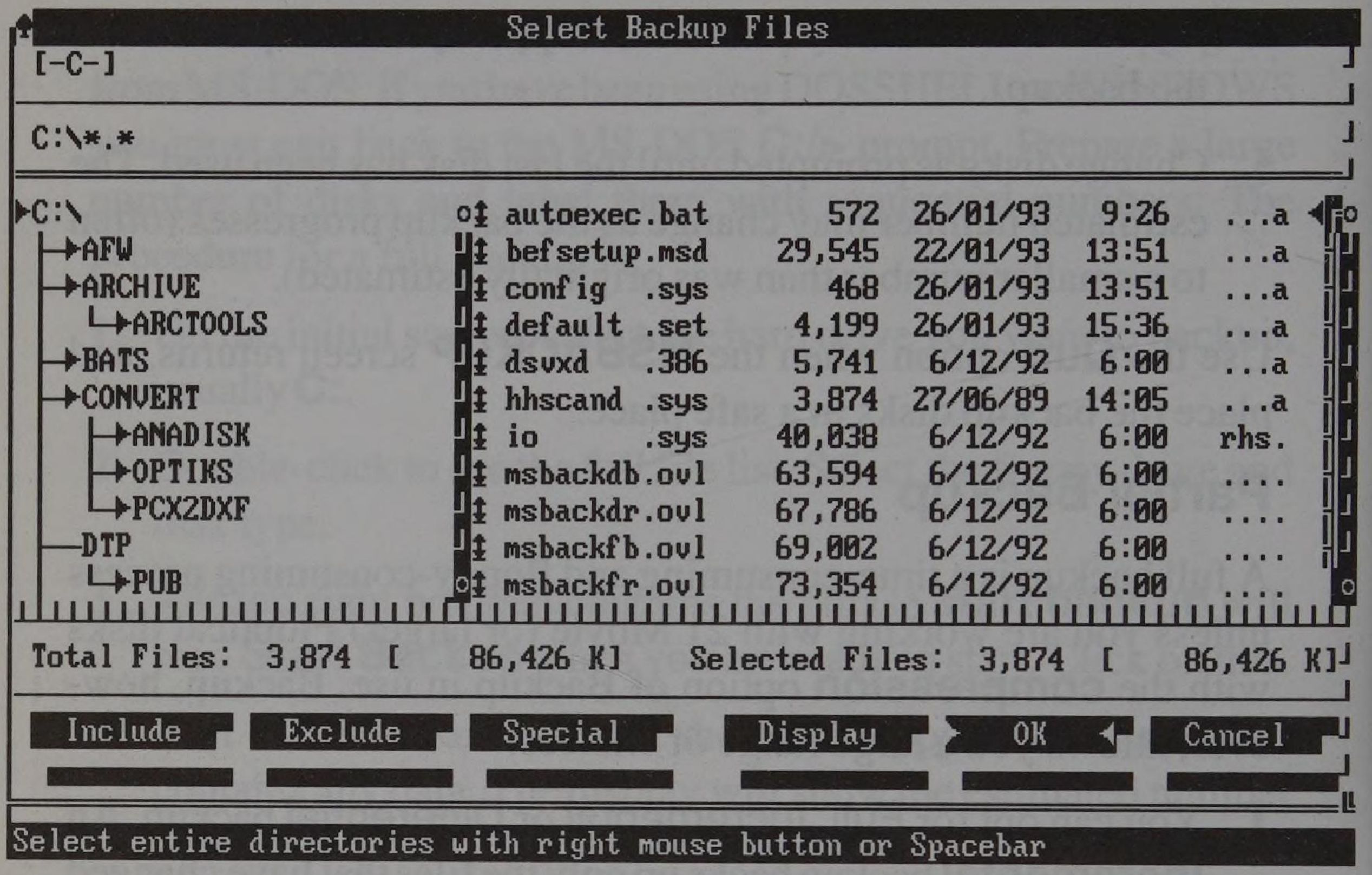
1. You can opt for Full, Incremental or Differential backup. An **Incremental** backup backs up only the files that have changed since the last backup (full or incremental). The archive bit for each file is reset.

- Use this if you have made a backup before and you still have these disks.

2. A **Differential** backup backs up all files that have changed since the last full backup, plus any other files you select, but does not change the archive bits, so that the next differential backup will save the same set of files plus any others that have changed.

3. You can also opt to save all marked files, and when you take this option you are presented with a file list, (illustrated in the screenshot on the next page), on which you can tick files to be saved.

58 Making Backups

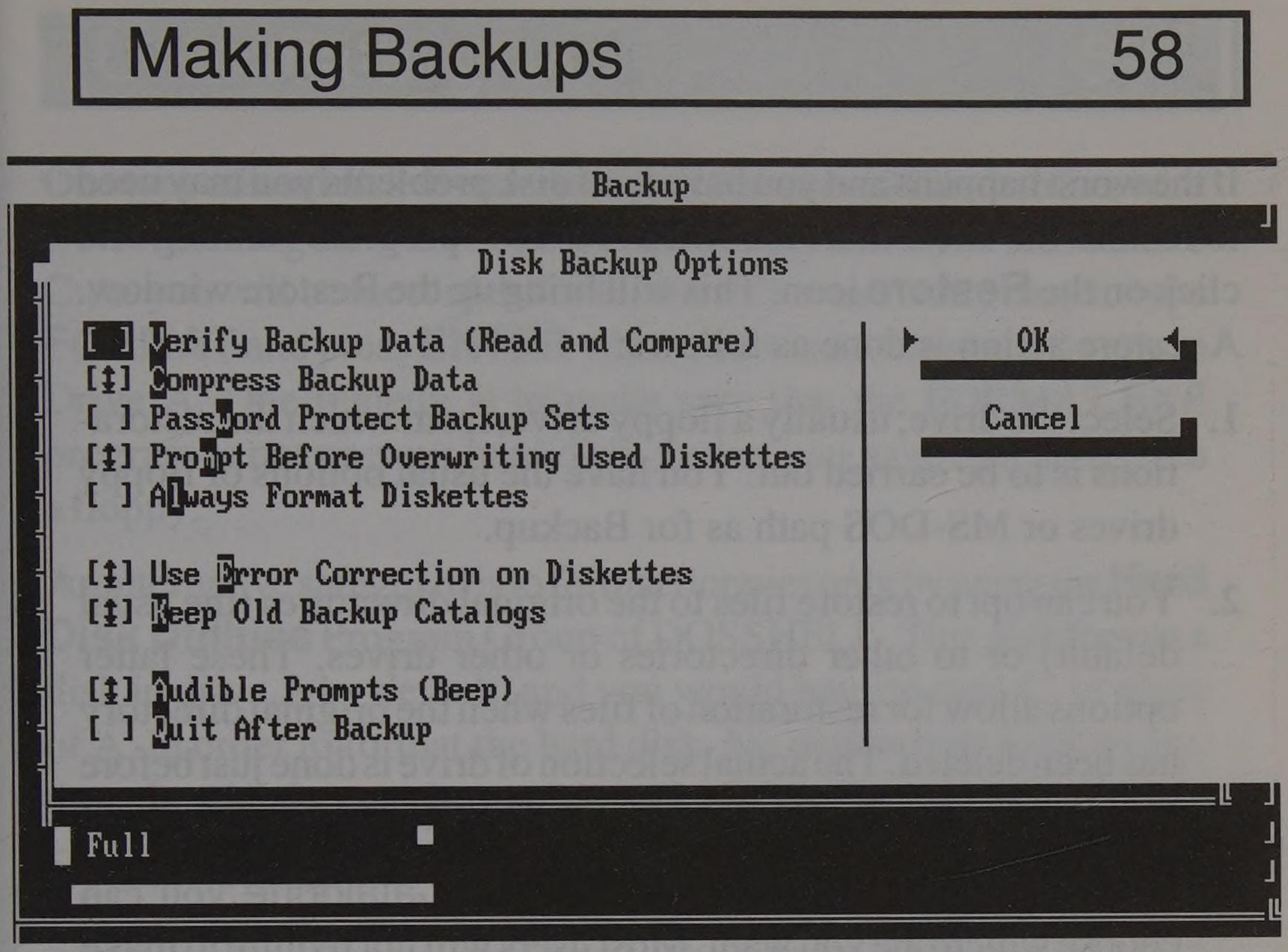


The file list which allows you to select files for selective backup.

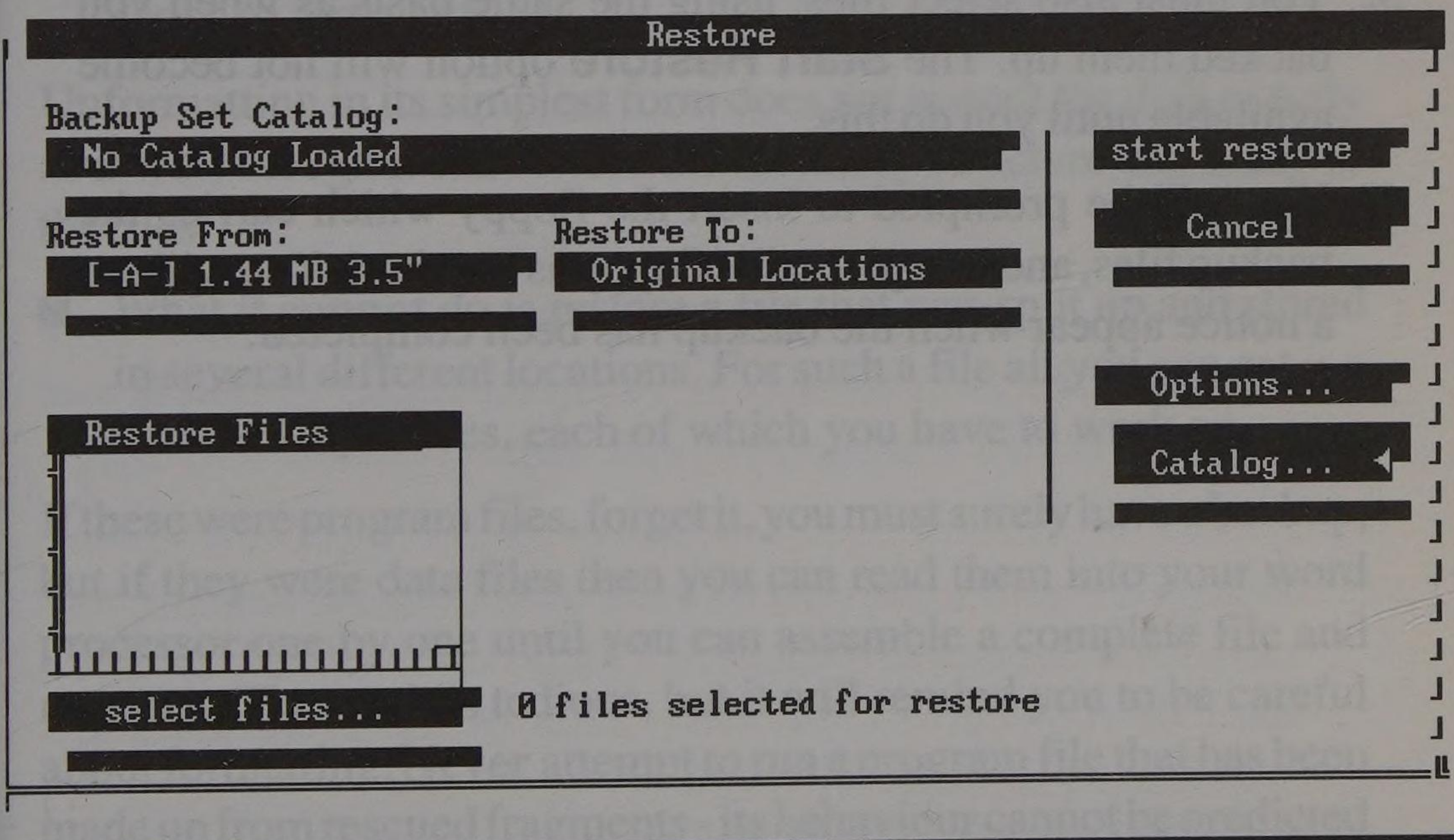
4. The **Special** option allows you to backup files in a specified date range, or to exclude files such as Copy Protected files, Read-only files, System files or Hidden files.

5. The **Display** option allows you to sort files by Name, Extension, Size, Date or Attribute, or to use wildcards to describe a set of files such as *.PCX.

The Options are illustrated below - avoid password protection unless you are certain that you can remember a meaningless password (if it means anything to you then it can be guessed by someone else).


Backup
Disk Backup Options
[ ] Verify Backup Data (Read and Compare)
[‡] Compress Backup Data
[ ] Password Protect Backup Sets
[‡] Prompt Before Overwriting Used Diskettes
[ ] Always Format Diskettes
[‡] Use Error Correction on Diskettes
[‡] Keep Old Backup Catalogs
[‡] Audible Prompts (Beep)
[ ] Quit After Backup
OK
Cancel
Full

The Backup options - the defaults are sensible but your own needs might require you to alter some of these options.



The window that appears when Restore has been selected

If the worst happens and you have hard disk problems you may need to restore the files. Start the **MSBACKUP** program running and click on the **Restore** icon. This will bring up the Restore window. A restore action is done as follows:

1. Select the drive, usually a floppy drive, from which the restorations is to be carried out. You have the usual options of floppy drives or MS-DOS path as for Backup.

2. You can opt to restore files to the original directories (the usual default) or to other directories or other drives. These latter options allow for restoration of files when the original directory has been deleted. The actual selection of drive is done just before backing up, and the C:\ drive is used as a default.

- If there is more than one Backup Set Catalogue you can choose which one you want. Most users will not require to make this choice.

3. You must also select files, using the same basis as when you backed them up. The **Start Restore** option will not become available until you do this.

4. You will be prompted to insert the floppy which carries the backup files, and to replace this floppy as required. You will see a notice appear when the backup has been completed.

Unformatting 59

One type of hard disk disaster that is rather more difficult to live with is reformatting the whole of the disk by accident. If you are in Drive C:, as you will be most of the time, then it's remarkably easy to type **FORMAT** and press **ENTER** when you want to format a floppy in Drive A. One remedy is to make sure that the FORMAT.EXE program is not on your hard disk, so that you have to read it from a floppy.

An alternative safeguard is to format floppies only by using the **Hard Disk Utilities** Program Group of DOSSHELL. This will format a disk in drive A by default, and you would have to type C: in place of A: in order to format the hard disk. No matter how hard we try, though, mistakes can be made, particularly after midnight.

- The formatting of a hard disk makes all the files disappear from the directory, but this does not replace the data on the disk, only the root directory and FAT (File Allocation Table) data. You can therefore use an UNFORMAT utility that will give you some scope to recover files.

Unformatting in its simplest form does not restore the disk exactly as it was, though it can restore the directory structure and many of the files.

- What it **cannot** do is restore a file that was split up and stored in several different locations. For such a file all you can get is a set of 4 Kbyte files, each of which you have to work on.

If these were program files, forget it, you must surely have a backup, but if they were data files then you can read them into your word processor one by one until you can assemble a complete file and record it. The work is tedious, but it will remind you to be careful about formatting. **Never** attempt to run a program file that has been made up from rescued fragments - its behaviour cannot be predicted and is most unlikely to be correct.

59 Unformatting

MS-DOS 6.0 includes an **UNFORMAT** command and, if it is used correctly, the disk, hard or floppy, can be restored, subject to one condition.

- The format must **not have used FORMAT /U**, which is the unconditional format command. Using **FORMAT /U** totally destroys any information that previously existed, and should be used only on a new floppy disk, or a disk that has been reporting read and write errors and for which backups have been taken.

UNFORMAT can be used only under DOS, and you have been using Windows or DOSSHELL you **must** be certain that you have closed these shell programs down before you attempt to use UNFORMAT.

1. Type **UNFORMAT A: /TEST** first, with a floppy in the drive. This will show what UNFORMAT will do, but does not actually carry out the actions.
2. If you are working on a floppy drive you will be asked to insert a disk. The process can be a **very** lengthy one even for a floppy disk, so allow plenty of time (20 minutes or more for a 1.4 Mbyte floppy).

   If you see a message referring to **MIRROR** you can ignore it - MIRROR was the name of the file undelete system used in MS-DOS 5.0. MS-DOS 6.0 uses two systems called **Tracking** and **Sentry**. The Tracking system is the same as the MIRROR type of system.
3. When you have some idea of what **UNFORMAT** will do, you can use the command without the **/TEST** option. **UNFORMAT** must be followed by the drive letter of the drive you have accidentally formatted.
4. As UNFORMAT works it will show the directories of the disk.

- If you use the UNFORMAT command in the form

  UNFORMAT /L

  you will see the filenames as well as directory names when the disk is being unformatted. You can optionally use

  UNFORMAT /P

  to send all messages to the printer.

60 Undelete Actions

MS-DOS 6.0 offers three layers of protection against the accidental loss of files because of unintended deletion. The lowest level depends on the fact that when a file is deleted its disk directory entry is changed (the first character of the name is replaced) but the data of the file is not deleted. Other levels depend on keeping a file of information about deletions. There are separate MS-DOS and Windows versions of UNDELETE, but this book is concerned with the MS-DOS version only.

- You cannot expect any form of undelete other than Sentry to be useful if a large number of new files have been saved since you deleted a file, because the space is likely to have been re-used.
- That said, you can sometimes be lucky and you might feel that you have nothing to lose.

The default system of protection is the lowest level, **DOS**, but when you use **UNDELETE** as a command the highest level, **Sentry**, will be tried first, followed by the second level, **Tracking**.

1. The **DOS** method requires no preparation, no memory-resident program and no additional disk space. It is attractive if you maintain a good backup system and very seldom make an accidental deletion. It is also more suitable for un-delete actions on floppy disks.

- The higher levels of undelete are useful if your computer is shared with other users and unintended deletions are frequent.

2. The **Sentry** system requires a memory-resident program (which is installed in AUTOEXEC.BAT) along with a disk file (limited to a maximum size of 7% of your hard disk) and it retains deleted files for a limited time.
3. The **Tracker** system needs only the memory-resident program, but no disk space.

- Do not expect UNDELETE to deal with deleted sub-directories, or with files in deleted sub-directories.

Using **UNDELETE** with no parameters will start the Undelete action on the current drive and directory. For many purposes this is as much as you need. You can also follow **UNDELETE** with a drive letter.

Here is a typical screen output for the command **UNDELETE A:**. For each file that is found and reported as recoverable you are asked to press **Y** to confirm that you want to recover the file. You also need to supply the first character of the filename.

```
C:\>undelete a:

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: A:\
File Specifications: *.*

    Delete Sentry control file not found.

    Deletion-tracking file not found.

    MS-DOS directory contains    2 deleted files.
    Of those,    2 files may be recovered.

Using the MS-DOS directory method.

     ?PRDTOC  DOC     17223 12/06/92 20:29   ...A  Undelete (Y/N)?y
     Please type the first character for ?PRDTOC .DOC: A

File successfully undeleted.

     ?TPTOC   DOC     10525 13/06/92 10:01   ...A  Undelete (Y/N)?
```

A typical screen report on a floppy disk Undelete action that will use MS-DOS directory un-deletion.

You can specify by using a wildcard form of filename that UNDELETE is to restore only files of a particular specification. For example, you can use:

UNDELETE C:\PCXFILES*.PCX

to specify that only files in this directory and with the PCX extension letters should be restored. This is a much faster process than undeleting everything.

- You can add option letters to the **UNDELETE** command to force it to use only one of the methods, but this is very seldom necessary.

The more advanced protection options require you to add a line to the AUTOEXEC.BAT file, or to start the system by typing the line as a direct command. The most advanced system is **Sentry**, but this requires both hard-disk space and memory space, and the amount of hard disk space can be quite substantial, enough to store all your deleted files temporarily. For most users, the **Tracker** system is ideal.

1. You can install the Tracker system by using a line, in AUTOEXEC.BAT, such as:

UNDELETE /TC-200

which will keep track of deleted files up to a maximum of **200** on the **C** drive.

If you omit the number, Tracker will assign a default. The default number depends on the disk size and is 303 for disks of 32 Mbyte and over, less for smaller hard and floppy disks. The range of numbers that can be used is 1 to 999. You **must** specify the drive letter.

2. When this line is included in AUTOEXEC.BAT, tracking will be effective the next time you boot up the computer and delete a file.

3. When a file has been deleted, type **UNDELETE** when the **C:\>** prompt shows. You will see a screen output such as that below. The first letter does not need to be supplied this time, and you only need to confirm whether or not you want each file restored.

```
C:\>undelete a:

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: A:\
File Specifications: *.*

    Delete Sentry control file not found.

    Deletion-tracking file contains    2 deleted files.
    Of those,     2 files have all clusters available,
                  0 files have some clusters available,
                  0 files have no clusters available.

    MS-DOS directory contains    2 deleted files.
    Of those,     2 files may be recovered.

Using the Deletion-tracking method.

      ATPTOC   DOC     10525 13/06/92 10:01    ...A  Deleted: 27/01/93 10:13
All of the clusters for this file are available. Undelete (Y/N)?
```

A typical Tracker report on deleted files. You do not need to supply the first letter of the deleted file.

The **Sentry** system is higher in level in the sense that it uses more of your disk and memory resources and is almost impossible to defeat.

1. You install Sentry by using a line in the AUTOEXEC.BAT file such as:

   UNDELETE /SC

This example sets up Sentry for Drive C.

2. When files have been deleted, using the direct MS-DOS command **UNDELETE /DS** will produce a screen output of the form shown here (for a floppy).

```
C:\>undelete a: /ds

UNDELETE - A delete protection facility
Copyright (C) 1987-1993 Central Point Software, Inc.
All rights reserved.

Directory: A:\
File Specifications: *.*

    Delete Sentry control file contains    2 deleted files.

    Deletion-tracking file contains    0 deleted files.
    Of those,    0 files have all clusters available,
                 0 files have some clusters available,
                 0 files have no clusters available.

    MS-DOS directory contains    2 deleted files.
    Of those,    0 files may be recovered.

Using the Delete Sentry method.

      ATPTOC   DOC     10525 13/06/92 10:01    ...A  Deleted: 27/01/93 10:17
This file can be 100% undeleted. Undelete (Y/N)?
```

A typical Sentry report on deleted files. Note that you get the other levels of reports unless you opt to use Sentry only.

3. You need to confirm whether or not you want to restore a named file.

Options

The **UNDELETE** direct command, particularly when either of its two higher levels have been installed, can be used with a number of options. You can use one of three option words between **UNDELETE** and the drive letter:

- **/LIST** will list the deleted files and indicate which can be recovered, but no recovery will be carried out. This can be used with all three levels of Undelete protection.

- **/ALL** will recover all deleted files without the need to confirm. This is best used when you are working in a specified directory and with a named extension. If Sentry or Tracker is in use the files will be recovered with the correct names, but if only DOS recovery is used, the program will supply a first character of #. If this would result in two files bearing the same name, other characters will be substituted in order of:

#%&123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ

- **/PURGE** is used only with Sentry, and will delete the contents of the Sentry file. If no drive is provided, **UNDELETE** will search for it, but you can speed the process by using, for example, **/PURGE :C**

You can also use option words following the drive/path information for **UNDELETE**. These are **/STATUS, /LOAD, /U, /S, /T**

- **/STATUS** will display what form of protection is available for a nominated drive.

- **/LOAD** will load the memory-resident part of the UNDELETE program, using information contained in a file called UNDELETE.INI.

- **/U** will unload the memory-resident UNDELETE program. This is essential if you want to change from one system to another while the computer is switched on.

- **/S** loads in the memory-resident part of the Sentry system and provides for creating a disk file. If the file UNDELETE.INI exists it will determine the extent of protection. The **S** letter can specify a drive or drives, using, for example, **/S:C**

- /**T** loads in the memory-resident part of the Tracker system and provides for creating a disk file. If the file UNDELETE.INI exists it will determine the extent of protection. The **T** letter can specify a drive or drives, using, for example, /**T:C** and you can also specify a maximum number of files (in the range 1 to 999) to track. For example, /**TC-200** will track up to 200 deleted files on drive C.

Remember that these options will usually be typed once only, into the AUTOEXEC.BAT file.

Note that Sentry creates a subdirectory called **SENTRY** that contains a file called **CONTROL.FIL**. Tracker creates a file called **PCTRACKR.DEL**, not in a separate sub-directory. Both files and the SENTRY subdirectory are hidden, and become visible if you use DOSSHELL with the option to see hidden files switched on. The SENTRY subdirectory and its files are **almost impossible to delete** and will survive even an unconditional **FORMAT**.

Disk and File Defragmentation 61

In the lifetime of a hard disk, you are likely to add and delete files many times over. Each time you delete a file, its space on the disk is released, and saving a new file will make use of some of this released space. A large file, however, may make use of many small amounts of space released by the deletion of many smaller files.

- This fragmentation results in much slower access times for such files, because of the number of times that the head of the disk drive has to be moved in order to read all the portions of a file.

MS-DOS 6.0 contains a utility called **DEFRAG.** This will reorganise the disk, compacting the scattered file clusters, caused by deletions, into blocks of continuous file and totally reorganising all of the data on the disk. As it operates, DEFRAG will save file fragments in memory and then store them into new positions on the disk.

- DEFRAG uses the word *contiguous* to mean files without gaps - it literally means that each part touches the next.
- You would expect that any power failure during this operation would cause loss of data and loss of the stored position numbers for the data fragments, but DEFRAG is claimed to be immune from this problem, as MEMMAKER is.
- There is absolutely **no chance** of recovering any deleted file after a disk has been defragmented, because the space that they occupied will have been re-used.
- Do not use DEFRAG on your hard disk if you have used DBLSPACE to compress data on to the hard disk, see Section 62.

If you have any doubts about disk defragmentation, make sure that your disk is fully backed up. In particular, keep a system floppy disk ready in case you get a message about the System files being

unavailable (**Non-System Disk** or **Disk Error**). You may have to alter your CMOS RAM settings if you have set them to avoid booting from a floppy.

- If you cannot boot from the hard disk after carrying out a DEFRAG, re-install MS-DOS 6.0 from the original disks. Trouble is usually due to the use of memory-resident programs during the DEFRAG action.

DEFRAG should be used when you have deleted a large number of files, or when you become aware that disk actions are taking longer and requiring much disk activity. The action should be used with caution, and with some preparation.

1. Make certain that you quit Windows or DOSSHELL, and try to ensure that no unnecessary memory-resident programs are running.

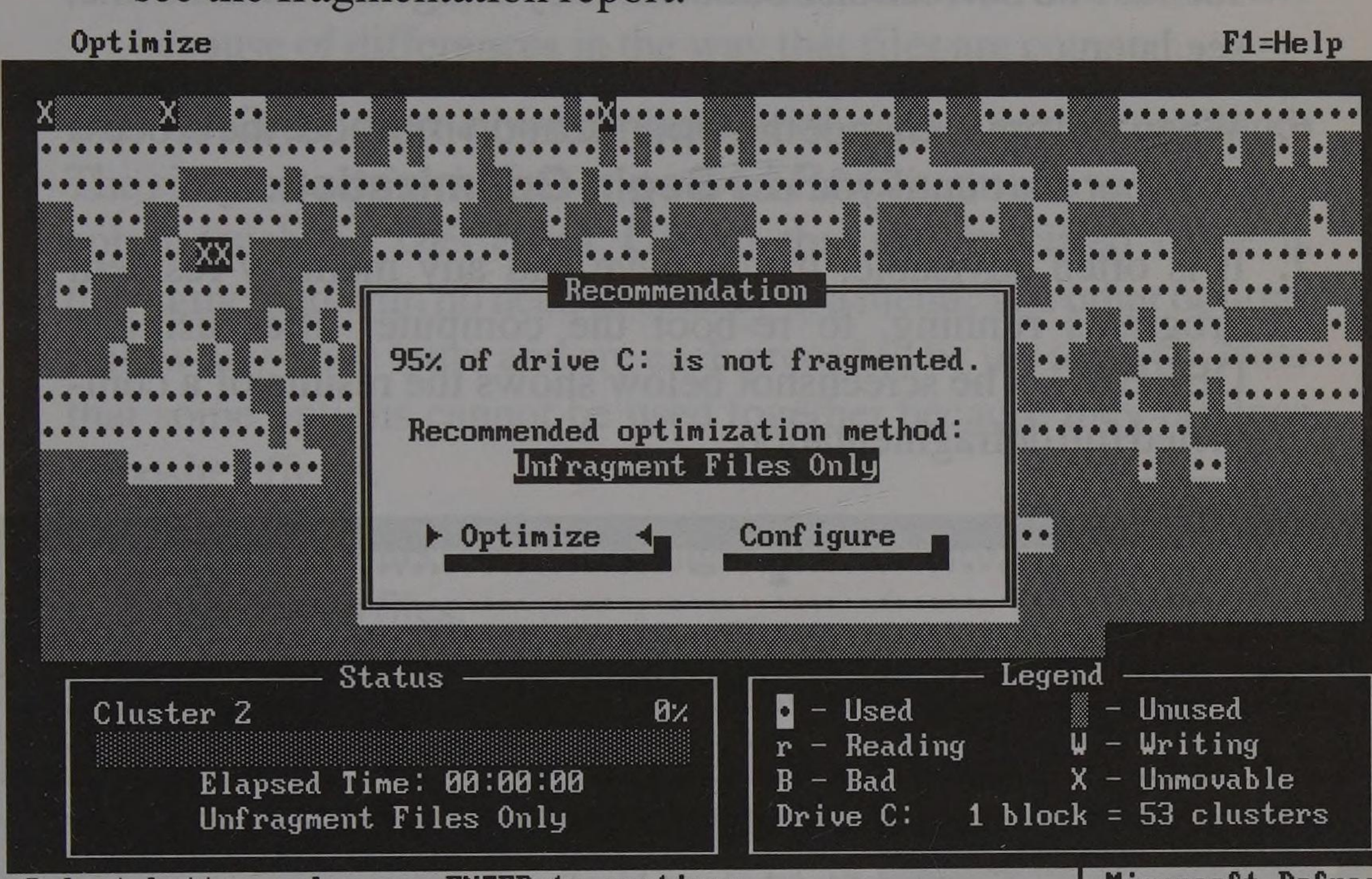
- You can reboot, pressing the **F5** key at the **MS-DOS Loading** prompt to ensure that no memory-resident software is installed. Remember that this will prevent any PATH from being established, so that you will have to find the directory that contains DEFRAG, usually by typing **CD \MSDOS**.

2. You should use the command **CHKDSK /F** first to ensure that there are no detached pieces of files on the disk. Wait until this action is complete and any file fragments either deleted or converted into complete files.

3. Type **DEFRAG** (press **ENTER**). The system will (briefly) be checked, and you are asked to select which drive to defragment (optimize). You will normally select **C:** - there is usually little point in defragmenting a floppy disk.

- The exception to this is a floptical disk of 21 Mbyte or greater capacity.

4. The disk information is read and directories listed. You will then see the fragmentation report.



A fragmentation report showing that 95% of the files are not fragmented. The display, however, shows that there are gaps between files so that the disk space is inefficiently used.

5. Click to start the action. The process of shifting files will start, and the display illustrates what is being done.

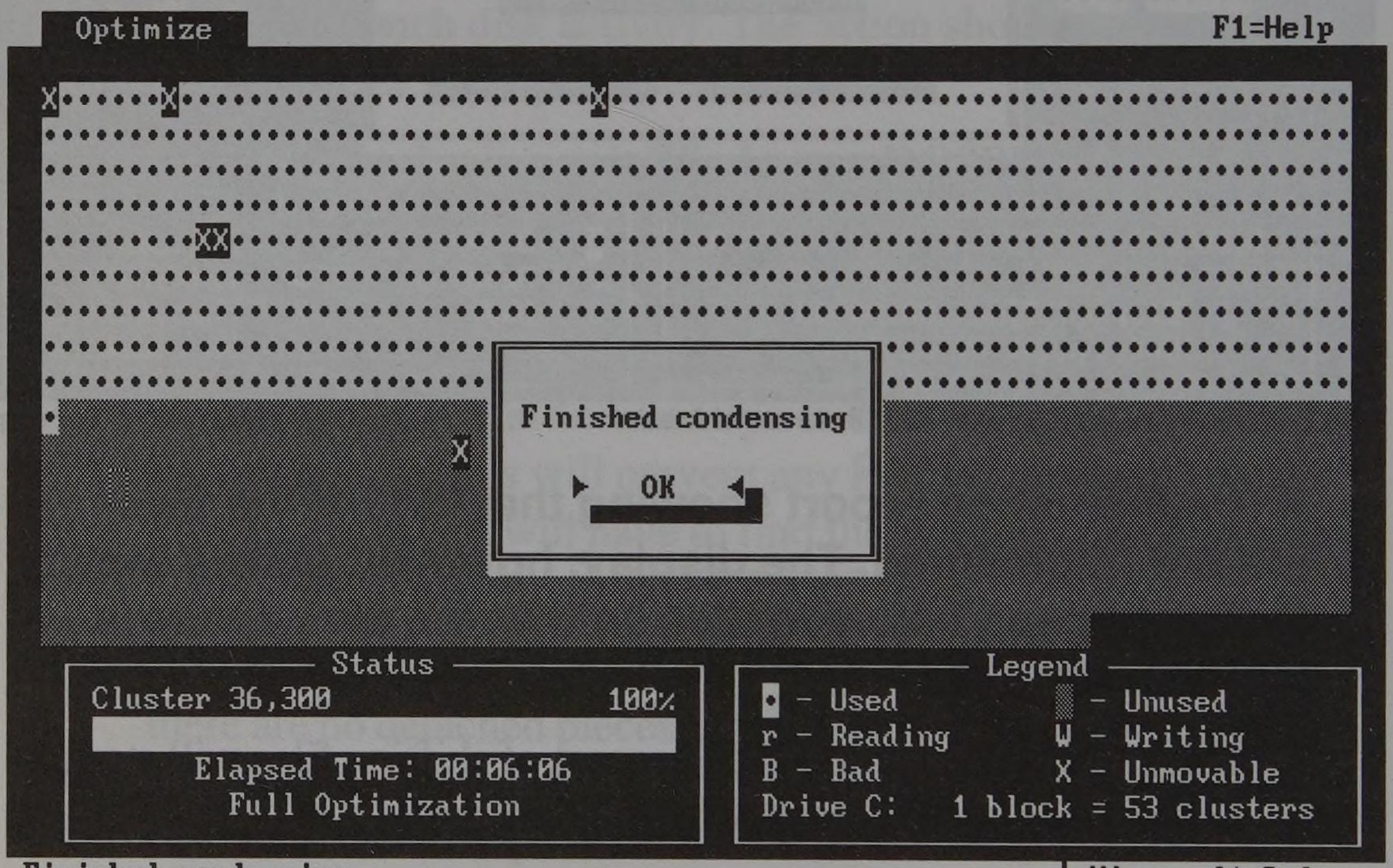
- If you do not want to use the recommended method, which in this example will defragment files but leave spaces between files you can click on **Configure** to select a method for yourself. This provides the options of Begin Optimization, Drive selection, Optimization method, File Sort, Map Legend.

  Full Optimization, the usual preference, takes considerably longer than the Unfragment Files only option.

- Note that you can leave DEFRAG by pressing **Alt-X**. Options such as File sort can also be obtained by using the command line, see later.

6. When you opt to change the Optimization method, the process does not start until you use **Begin Optimization**.

7. It is often advisable, if you have had any memory-resident programs running, to re-boot the computer after using a DEFRAG. The screenshot below shows the results of a completed full defragmentation.



The results of a full defragmentation as they are shown on the screen.

The DEFRAG program is normally loaded into Upper memory if this is available, and the recommended de-fragmentation process will depend on the result of checking the drive - the illustration shows a drive on which a large number of files have been deleted but with only a small percentage of the disk fragmented.

- DEFRAG cannot be used over a network or over INTERLNK. The DEFRAG reports are not identical to those from CHKDSK because of differences in the way that files are counted.

DEFRAG can be used with a number of options in the command. The simplest option is the follow **DEFRAG** with a drive letter and colon (such as **DEFRAG C:**) so that the specified drive is defragmented with no need to specify in a menu. The other options follow any drive letter and are each preceded by the slash (/). Note that some options cannot be used together because they produce opposite effects.

/F Defragments files and arranges files so that there are no spaces between files.

/U Defragments files, but leaves any empty spaces that occur between files.

/S Defragments files and uses a sorting order determined by the following letter which must be separated by a colon or a space. In the following list the use of the colon is illustrated.

:N Alphabetical order of name

:N- Reverse alphabetical name order

:E Alphabetical order of extension

:E- Reverse alphabetical order of extension

:D Order of date and time, earliest first

:D- Order of date and time, most recent first

:S Order of size, smallest first

:S- Order of size, largest first

/B Causes the machine to re-boot after defragmentation is compete.

61 Disk and File Defragmentation

/V Verifies that files have been written. This slows down the DEFRAG action and does not confer any particular protection on the writing process.

/SKIPHIGH Runs DEFRAG from conventional memory, taking up space that otherwise is used for storing files, and so slowing down the action.

/LCD Uses DEFRAG in a form move visible on an LCD display.

/BW Uses DEFRAG in a form more visible on a monochrome screen display.

/G0 Disables the mouse and graphics character set for the DEFRAG display - use only if you have no mouse.

Using DBLSPACE 62

DBLSPACE is a program that makes better use of disk space, and is particularly useful for the hard disk. All normal files contain redundant portions, and by compressing the files to eliminate this redundancy the amount of disk space needed by a file can be greatly reduced.

- The amount of time needed to compress the file is recovered because the time needed to load or save the smaller file is less.

DBLSPACE can be loaded in as the machine boots up and it remains in memory, intercepting each save and load to or from the hard disk. It compresses files that are being saved, and expands files that are being read. All of this is automatic and requires no attention from you.

- The use of DBLSPACE can, on average, double the amount of data that can be stored on a hard disk. Some files can be only slightly compressed, others by a greater factor. On a machine that I use for text, the factor was around 1.8. On a machine used mainly for graphics, the factor was 2.0. Some files can be compressed more than this.

There have been various space-releasing programs available before DBLSPACE, but these cannot be guaranteed to be perfectly compatible with MS-DOS.

- When a drive has been established with DBLSPACE, it must be used with DBLSPACE utilities for undeleting and defragmenting. You should not attempt to use the normal UNDELETE or DEFRAG on a DBLSPACE drive, and special DBLSPACE versions of these utilities are supplied as part of the DBLSPACE program.
- Floppy drives can be double-spaced, but this requires you to use a MOUNT and UNMOUNT procedure, rather like a format, on each doubled disk **each time you use it or change disks** as well

as an initialisation for each doubled disk. If you do not exchange disks with other people, you can place these necessary commands in the AUTOEXEC.BAT file and gain the advantage of using floppies with a much greater file capacity. Even if you sometimes need to prepare a disk for other users, you can always turn off the Doublespace action easily and use the drive with a normal disk.

- You cannot use DBLSPACE on floppies if you work from DOSSHELL unless you have turned off multi-tasking.

If you decide to convert your hard disk to doublespacing, you should ensure that the drive is fully backed up. If you have a tape streamer or Floptical system, use it rather than backing up on ordinary floppies.

Before you install DBLSPACE, **make sure** first of all that you have fully escaped from any form of Shell program such as Windows or DOSSHELL. Ideally, you should not have **any** memory-resident programs running.

1. You are offered the choice of **Express Setup**, the default, or **Custom Setup**. Custom Setup is useful only for unusual circumstances, and you need to be more knowledgeable about your hard disk system in order to use it.
2. To proceed with **Express Setup**, press **ENTER**. This will create the doublespaced disk and convert all of your existing files. The procedure can be lengthy for a large hard disk.

- Note that if the disk has become almost full it will be impossible to use DBLSPACE. Back up some large files and then delete them and defragment the disk before trying again.

3. You will see a time estimate - it was 50 minutes for my 130 Mbyte hard disk. You are warned that the machine will be re-started in the course of the conversion.

4. You are asked to press the **C** key to start the doublespace conversion. The time estimate that is shown on the screen may alter in the course of conversion - don't be surprised if it promises completion in 33 minutes and then alters this to 40 minutes, because the time needed depends on the structure of the files and this is unknown until compression is started.
5. **CHKDSK** is run automatically before the work starts. The computer re-starts and the system is checked. What is happening is that files on the disk are being compressed into one large file called DBLSPACE.000 that is held on the hard disk, but with a different drive reference letter (**H** in my example).
6. The C: drive can still be used as it was before and shows the normal directory. Loading a file from the C: drive now causes the file to be extracted from its packed form, and saving a file to the C: drive packs the file in again.

- The C: drive is referred to as the *Compressed* drive, and the other (imaginary) drive as the *Host* drive.
- A few files are left unaltered, notably the MSDOS.SYS, IO.SYS and COMMAND.COM files, and any permanent swap file used by Windows. These are placed on the Host drive, the part of the hard disk that is used also to store the compressed file. There will be some spare (uncompressed) space remaining

If you use DOSSHELL, however, you can opt to make hidden files visible, and if you do this it is possible to delete the DBLSPACE.000 file. You risk losing virtually all of your data if you delete hidden files, so you should **never work with the DOSSHELL option to make hidden files visible** unless it is essential to do so.

Custom Setup is used for special requirements, such as keeping half of a hard disk uncompressed, to compress a different hard disk

(not the C: drive), or to decide for yourself what drive letter to allocate to the Host, or what estimate of compression ratio to use. You also have a choice of compression methods. Custom Setup is for the experienced user only.

Using the double-spaced drive

Once the hard drive is double-spaced, you use your C: drive exactly as you have used it before. You must not attempt to use the H (or whatever letter has been assigned) drive. Because your data now exists in the form of a huge single file, the normal CHKDSK and DEFRAG utilities are meaningless, and you should not attempt to use them either.

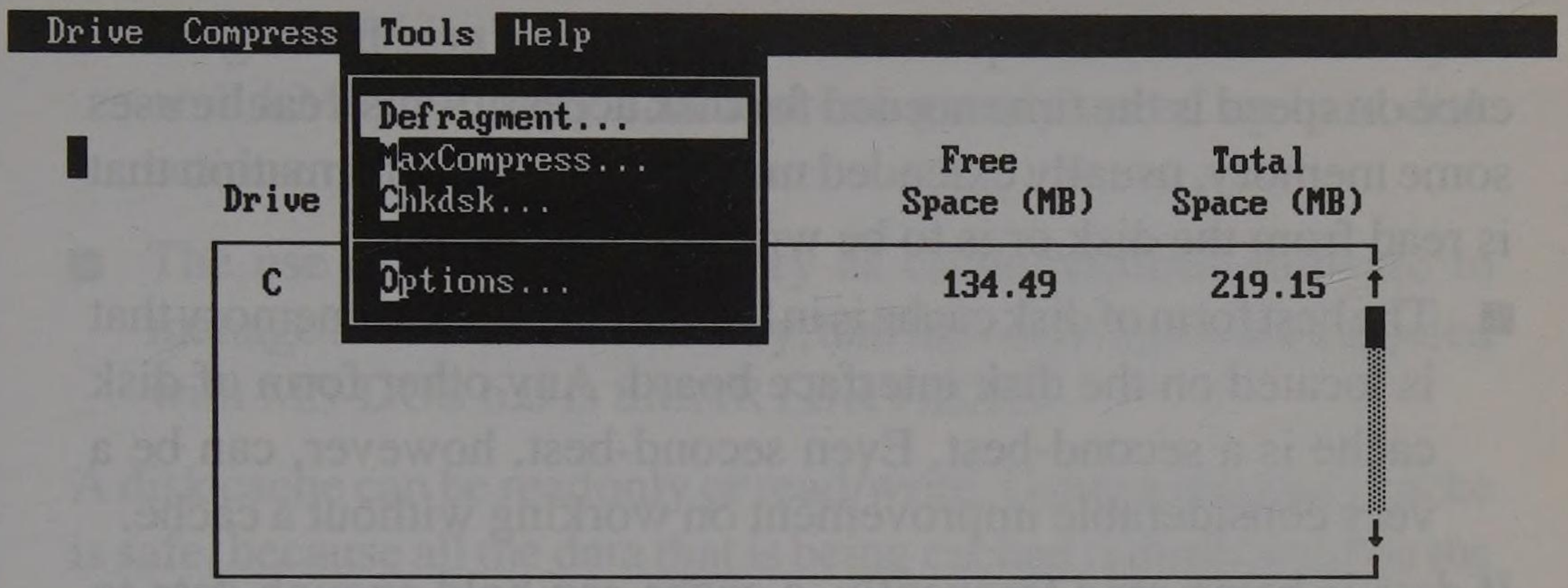
These actions are now available from DBLSPACE, either from the main menu of DBLSPACE or by way of commands. To use these actions from the menu:

1. Leave DOSSHELL or Windows completely if you have been running either of these Shell programs.
2. Type **DBLSPACE** (press **ENTER**). You will see the **DBLSPACE** menu screen.
3. Click on **Tools** to see the menu.
4. Select the action that you want, usually **Defragment** or **Chkdsk**, either by double-clicking with the mouse or moving the cursor and pressing **ENTER**.

The alternative is to carry out these actions by using commands from DOS directly by using option words following a **DBLSPACE** command.

To defragment type:

DBLSPACE /DEFRAGMENT


Drive Compress Tools Help
Defragment...
MaxCompress...
Chkdsk...
Options...
Drive
Free Space (MB)
Total Space (MB)
C
134.49
219.15
To work with a compressed drive, press the UP ARROW or DOWN ARROW key to select it. Then, choose the action you want from the Drive menu.
To quit DoubleSpace, choose Exit from the Drive menu. For help, press F1.
Consolidates free space on a compressed drive

The DBLSPACE menu as it first appears, showing the main menu and the Tools menu which contains the DEFRAG and CHKDSK utilities for compressed drives.

To check the compressed file use

DBLSPACE /CHKDSK

You can also use other DBLSPACE options by adding parameters to the command. See the *Pocket Book of MS-DOS 6.0* for details - some of these options need to be used with care.

63 Using theSMARTDRV Cache

For a machine with adequate memory, the most constricting influence on speed is the time needed for disk access. A disk **cache** uses some memory, usually extended memory, to hold information that is read from the disk or is to be written to the disk.

- The best form of disk cache is in hardware, meaning memory that is located on the disk interface board. Any other form of disk cache is a second-best. Even second-best, however, can be a very considerable improvement on working without a cache.

If data is being read frequently, a cache can hold enough data to ensure that the data can usually be supplied from the cache rather than directly from the disk. For frequent writes, the cache can hold data until there is too much to hold in memory and a disk write becomes essential. In either case, the cache is the main reservoir of data, and the computer reads from and writes to the cache if possible.

Because the cache is memory, reading and writing can be fast and efficient, very much faster than reading from or writing to a hard disk. In addition, a good cache system will make use of the disk drive at a time when nothing much else is being done, such as when the machine is waiting for you to press a key. In this way, disk speed has very much less effect on the overall speed of the system.

If the cache is being used for writing and the cache fills up, the content of the cache is saved to the disk. If the cache is being used for reading, the first request for data will cause the hard disk to be read and to read enough to fill the cache. Subsequent reads will be from the cache unless the data required is not in the cache, which will cause the disk to be read again. A small cache for reading is not efficient if small amounts of data are read from scattered parts of the disk, but if the cache is large it may hold almost all the contents of a directory.

If extended memory is used for à cache, a large amount of memory can be devoted to the cache, typically 1 Mbyte.

- A cache of this size will hold most of the contents of a 1.2 Mbyte or 1.4 Mbyte floppy disk and can greatly speed up hard disk actions.
- The use of extended memory as cache requires software to manage the use of the memory, and the cache software supplied with MS-DOS 6.0 is SMARTDRV.EXE.

A disk cache can be readonly or read/write. Using a readonly cache is safe, because all the data that is being cached is duplicated on the disk, and if anything should happen to cause a reboot (a hiccup in the power line, for example) no data will be lost from the cache that cannot be replaced.

A read/write cache, by contrast, carries the risk that there will be some data in the cache that has not been written to the disk, though as far as the program you are using is concerned, it has been saved. A reboot that is carried out for any reason will result in this data being lost. The greatest risk is that if the machine locks up with a program running, data that you think has been saved will not have been written on the disk.

Cache software that will cache writing as well as reading usually tries to get around this hazard by writing data to the hard disk within a 5 second deadline whenever there is a period in which the disk is not being used, and they also provide for a write command that can be used to force a write if you want to reboot and are not certain that the cache has written data.

No safeguards, however, will prevent data from being lost if you switch off abruptly, or suffer a power cut or a machine lockup, and if data is very precious it is probable better to disable write caching. SMARTDrive and other cache programs allow this to be done by using an option letter in the command that starts the cache running.

63 Using the SMARTDRV Cache

SMARTDRV.EXE is used in the AUTOEXEC.BAT file and is set up as follows:

1. Start an editor program and type as the filename
   C:\AUTOEXEC.BAT

- If you have Windows 3.1 installed, its Setup will already have placed the SMARTDrive line at the start of the AUTOEXEC.BAT file.

2. At any place in the file, preferably near the start, add the line:
   SMARTDRV
3. Save the file again.

This assumes that you are using a machine with adequate extended memory, and that the extended memory manager HIMEM.SYS is installed in the CONFIG.SYS file ahead of SMARTDRV.

- A few memory-resident programs object to SMARTDRV being placed ahead of them, and you may need to move its position in the AUTOEXEC.BAT file if there are problems after installing SMARTDRV.

If you encounter problems with SMARTDRV, such as slow action, odd reports (like **Packed File Corrupt when using HELP**) this is usually due to conflicts in high memory. Try first running without SMARTDRV - this should instantly cure the problems and the odd messages. Then try running SMARTDRV in conventional memory. If all is still well, it points to a problem in the EMM386 line in CONFIG.SYS. This type of problem usually occurs after MEMMAKER has been used, and the culprit is usually a HIGHSCAN addition in this line.

By using only the command name, you allow SMARTDrive to select automatically the optimum operating conditions. These will depend on how much extended RAM is present and whether or not you use Windows.

The defaults are:

Available Extended RAM	Used by SMARTDRV:	Reduced by Windows to:
1 Mbyte or less	All	Nil
1 to 2 Mbyte	1 Mbyte	256 Kbyte
2 to 4 Mbyte	1 Mbyte	512 Kbyte
4 to 6 Mbyte	2 Mbyte	1 Mbyte
More than 6 Mbyte	2 Mbyte	2 Mbyte

Note that if your machine has only 1 Mbyte or less of extended RAM (a machine with 1 Mbyte total has only 384 Kbyte of extended RAM) then SMARTDrive will have no effect on Windows, since all of the available extended RAM will be used directly by Windows. For SMARTDrive to become really effective, the total extended RAM should be 4 to 6 Mbyte.

A cache of more than 2 Mbyte is not really much more useful than one of 2 Mbyte, and any gain from using more memory than this is not commensurate to the cost of the memory. You should not consider using a RAMDrive unless SMARTDrive is able to make full use of memory and you have an application that needs RAMdrive.

64 Options for SMARTDrive

SMARTDrive contains a wide range of options. If SMARTDRV.EXE is in a directory other than the root directory, its path must be shown in the command line. If, for example, the SMARTDRV.EXE files are in a directory called MSDOS, the command will be used in the form:

C:\MSDOS\SMARTDRV

The three main options are to show which drives will be included, the initial (main) cache size, and the Windows cache size.

By default, SMARTDrive will cache **all** of your drives other than CDROM and network drives. Floppy disks are by default cached for read only, and hard drives are cached for both read and write. You can exempt floppy drives from cache use by placing a minus sign following the drive letters, so that:

SMARTDRV A- B-

will ensure that drives A and B are not cached at all. The hard drive C will be fully cached for both reading and writing. The drive letter can be used alone, with the **+** sign or with the **-** sign. The drive letters are used as follows:

C used alone would cache this drive for reading only.

C+ would cache this drive for reading and writing.

C- would exempt this drive from cache use.

If you omit the drive letter **C** completely, the C drive will be fully cached - equivalent to using **C+** in the SMARTDrive line.

The use of numbers for Initial cache size and Windows cache size is unnecessary if the default values shown in the table above are satisfactory, as they usually are. Note that the values of memory shown in the table are of *available* extended memory, which is not the same as total memory or the amount of extended memory that

has been added. For example, if you have added 1 Mbyte of extended memory and have used only a small amount of the 384 Kbyte left over from the first Mbyte of RAM, the default values for SMARTDrive are 1 Mbyte of initial cache, with Windows able to take all but 256 Kbyte when it needs extended memory.

The other options should not be used unless you have enough experience with SMARTDRV to know what the effect will be, or if you are instructed by Microsoft (in a READ.ME file, for example) or by notes in any of the PC magazines. The only essential options are listed below.

1. **/C** forces SMARTDrive to write all cached data to the hard disk. This option would be used in a direct command, so that if you suspected or feared that data had not been written to the disk you would type:

   SMARTDRV /C

   to enforce the writing of any such data.

   This is seldom necessary, because writing takes place when there is no other disk activity. Use the **/C** option only if you are going to switch off or reboot abruptly and you are uncertain whether data has been written or not. You might also want to use this command at the end of a batch file that has started a program running, to ensure that all data is written before starting anything else.

2. **/S** is used while SMARTDrive is in use, in the form of the command:

   SMARTDRV /S

   to obtain information on the success of SMARTDrive. A typical display of the **/S** option is given overleaf, showing the number of hits (when SMARTDrive has supplied data without the need

to read the disk) and misses (when the disk had to be read). The status of drives is also shown. This illustration has been taken on a small-memory system soon after booting, so that the proportion of misses is unusually high.

```
Microsoft SMARTDrive Disk Cache version 4.0.091
Copyright 1991,1992 Microsoft Corp.
Room for 120 elements of 8,192 bytes each
There have been 1,198 cache hits
and 1,051 cache misses
Cache size: 983,040 bytes
Cache size while running Windows: 0 bytes
Disk Caching Status     drive  read cache write cache buffering
                          C:       yes         no         no
For help, type "Smartdrv /?"
```

A display from the SMARTDRV /S command, with an unusually high proportion of misses.

3. **/?** is used in the command:

   SMARTDRV /?

   to display a summary of Help on SMARTDrive and its options.

- 7 -

Viruses and Interlink

65 Viruses

Hard disk viruses are a menace that affect very few hard-disk users, but which, like some other crimes, cause more worry and expense than real damage.

- A virus is, strictly speaking, a piece of code that can attach itself to a program and copy itself when the program runs. This enables the code to be transmitted to other programs and also from one computer to another. We use the term also of other unwanted codes that can be loaded into a computer and which from then on will cause problems to appear, whether these can be spread to further computers or not.

The main types of virus are

1. The **Boot Sector** type that locates itself in the boot sector of the disk and loads in when the computer is booted.
2. The **Parasitic** virus, which is attached to a program file and is activated when that file is used.

- No problems have yet been reported of a virus infecting CMOS RAM, so that all viruses die when a machine is switched off.

3. The **Trojan Horse** is not really a virus, but it can cause problems that are just as serious. It takes the form of a program with an interesting title. When the program is loaded and run, it carries out the damage.
4. The **Worm** is also not really a virus but a piece of code that reproduces itself continually within the computer that it affects until so many copies have been made that there is no further room on the hard disk. It also has a serious effect on networks.
5. **Bombs**, also not necessarily viruses, come in two forms, **time bombs** and **logic bombs**. A bomb program, once loaded, saves itself on the disk and does nothing until some condition is met.

A **time-bomb** will be activated by date (like Friday 13th) or time (like midnight) and will carry out its action if the computer is running at this time.

A **logic bomb** will operate when other some conditions are met, such as 65% or more of the disk being used, or a copy of some well-known program being installed.

The true virus will attach itself to programs, and if these programs are copied to other computers this will allow the virus to be spread. The problem is not one that affects machines that run only floppy disks, because a disk with a virus can be thrown away.

- You can hardly throw away your hard disk if it is infected.

In addition, viruses are not effective on data files, because data files are not composed of instructions, so that the virus instruction cannot be executed. The data files can, however, become corrupted by the virus.

A virus can affect your computer **only** if you load and run a program from a disk that contains the virus, or load software of unknown origin over the telephone lines by way of a modem. Millions of computer users who do not use a modem and are careful about where they buy software are at no risk of virus infection.

- Bombs most commonly appear in commercial systems after a programmer has been sacked.

Many viruses are, like graffiti, comparatively harmless and have been devised by programmers wishing to demonstrate their skills. A virus may simply put up a message of the *Kilroy was here* variety on your screen, but at the other end of the scale it may cause files to vanish from your hard disk directory. This damage is not necessarily irreversible, but even if it is not it can take a long time to sort out.

65 Viruses

The really destructive viruses are fortunately rare on well-kept systems.

Three guidelines can help you to avoid virus problems:

1. Always be suspicious about unsolicited gifts of disks such as demonstration disks. If in doubt, try them on an old floppy-only machine.
2. Disks that have been run in a large number of machines are also suspect, as they could have picked up a virus from an infected machine.
3. Free disks issued with magazines are usually checked, but mistakes can happen, and it would be foolish to accept such a disk unless you had unwrapped it yourself from its original container.

There is no simple way of detecting a virus, particularly now that computer programs exist to generate new viruses. Do not think that because a **DIR** command shows program files of normal size they have not been altered. It is commonplace now for a virus that adds more code to a program to alter the program so that it reports the normal number of bytes.

The best method of dealing with viruses is not to become infected. More practically:

1. Beware of gifts of disks. Obtain programs only from reputable suppliers, whether these are commercial programs or shareware. **Never** run games programs that are passed on from another user.
2. Beware of demonstration disks or system checking disks that have been inserted into a large number of computers.
3. Try out suspect disks on a floppy-only machine. There's a place for that old Amstrad PC1512 after all.
4. Keep adequate backups of all program and data files on floppy disks. In particular, make several floppy System disks when you first install MS-DOS 6.0. Write-protect these floppies and keep them in reserve.
5. Do not leave your PC running unattended if there are people around who might be tempted to try a floppy disk on it.
6. If your **shared** PC permits password protection, use it.
7. Install the MS-DOS 6.0 anti-virus software. This is really a last resort.

Virus diagnosis

One possible sign of a virus infection is disk activity when you would not expect it. This is not always easy to detect, because many viruses are introduced at the time of booting the computer when the hard disk is working anyway.

- In addition, several programs will make automatic backups at intervals, causing disk activity, and when you use SMARTDRV in read/write mode you will often have disk writes at quiet times.

Some viruses signal their presence by messages, by bizarre behaviour (letters on screen messages suddenly drop down the screen, one

by one) or by the appearance of graphics on a text-based program. If a program whose behaviour you know well by experience suddenly starts to behave in a peculiar way, virus infection is one possible cause. Deal with it as follows:

1. Shut down the system as soon as possible. Save your data in the usual way - if you are worried about the data being in good condition, use a fresh floppy disk to save the data and label this disk so that you know it might be suspect.

2. After a few minutes, boot the computer from one of your write-protected copies of a System disk. This, unless you have been very careless or unlucky, should be free of any virus.

3. Re-install on to the hard disk the program that gave trouble, using a copy of the original floppies. Take the floppy out of its drive.

4. Run the program again from the hard disk.

5. If the trouble has ceased you may have eliminated the virus, but if it recurs after booting from the hard disk, try the preceding steps again, and following step 3 copy COMMAND.COM from a system floppy to the hard disk

6. If you have not eliminated the trouble, use the anti-virus software of MS-DOS 6.0 as described below.

- Anti-Virus software is not a cure-all. It will not necessarily detect all forms of virus nor remove them, so that your main efforts should always be directed to prevention rather than cure. Because new viruses are being written, any anti-virus system needs to be updated regularly.

The most important precaution is to keep a set of write-protected floppy disks that contain the MS-DOS System tracks with the IO.SYS, MS-DOS.SYS and COMMAND.COM files.

- These floppies should be prepared directly from the MS-DOS 6.0 installation disks, or on a machine with no hard disk, so that they can be guaranteed free from infection.

These can be used to boot up your computer if you think the hard drive may be infected. They are also valuable in the event that any damage occurs to the boot tracks of the hard disk.

67 MS-DOS Anti-Virus

The MS-DOS Anti-Virus system consists of software that will scan the hard disk (or a floppy) for signs of a known virus and which will report or destroy the virus. In addition, there is a memory-resident program that will detect when any program tries to carry out actions that are associated with virus activity. These are:

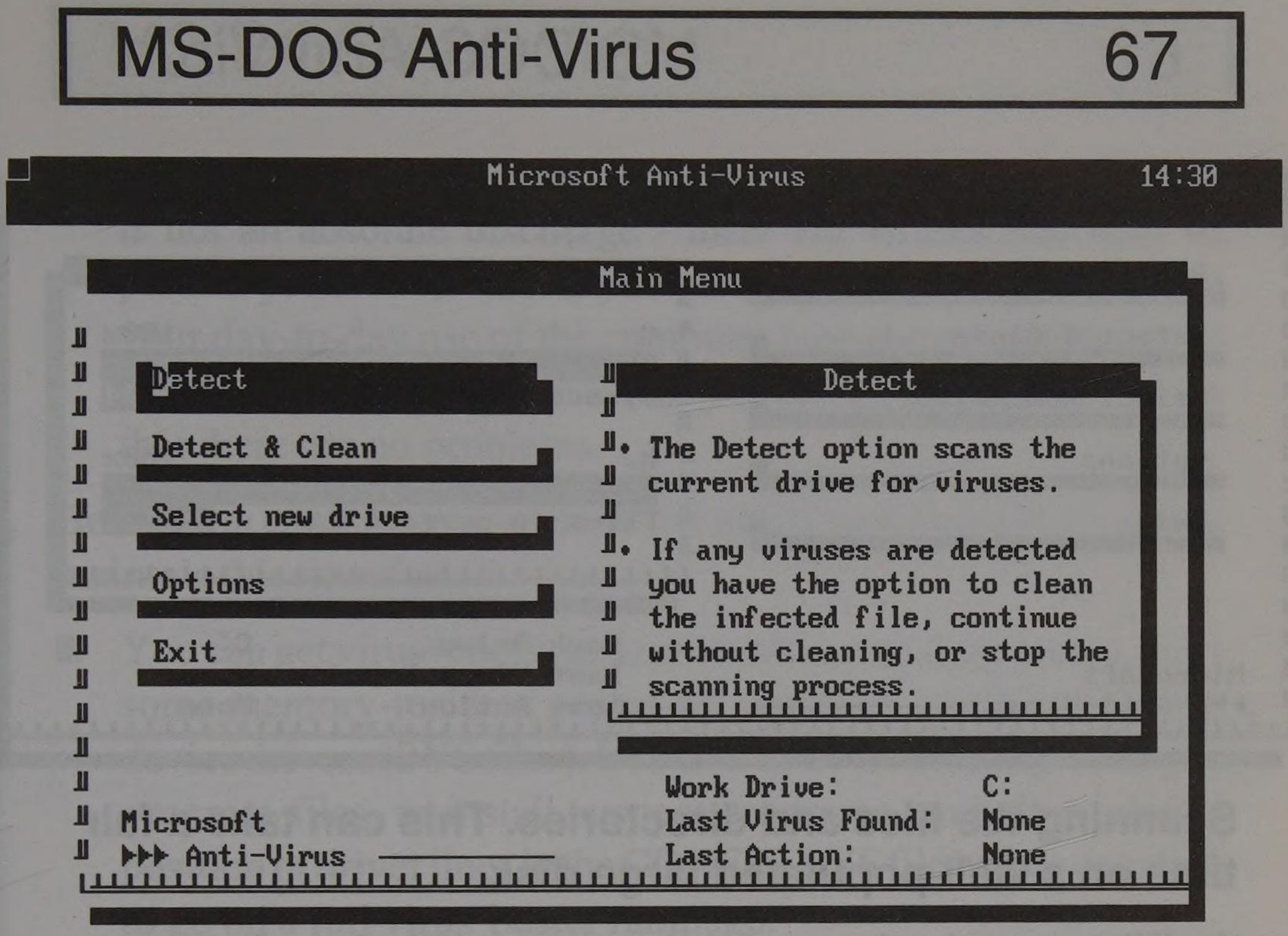
- Starting a low-level format, which totally erases any data on the hard disk
- Attempting to stay resident in memory using normal MS-DOS methods (other methods are not easily detected).

No program, other than FDISK, should ever carry out a low-level format, but there are many programs that will make themselves resident in memory (the VSAFE anti-virus program of MS-DOS 6.0 included), so that the second action should cause concern only if it is unexpected.

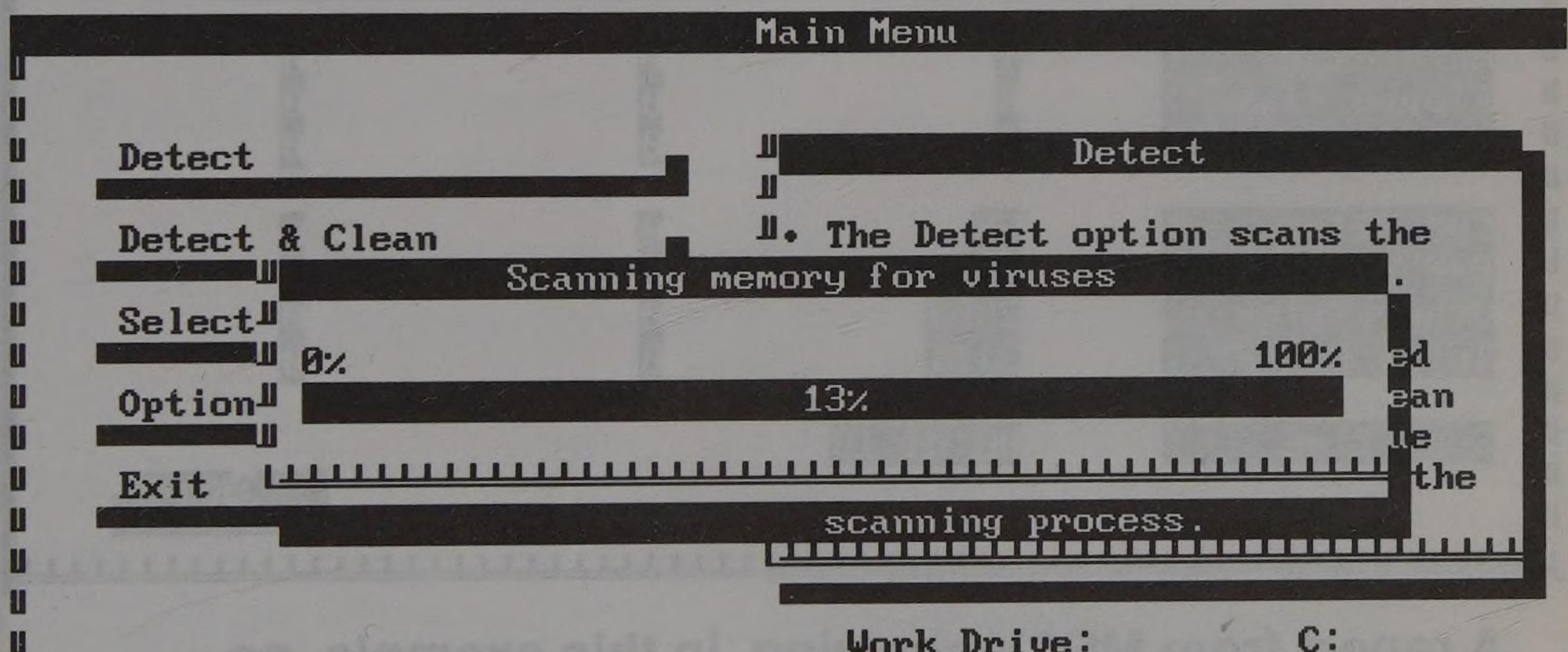
This software exists in two versions, DOS and Windows, and in this book only the DOS version is discussed.

To check for existing virus infection:

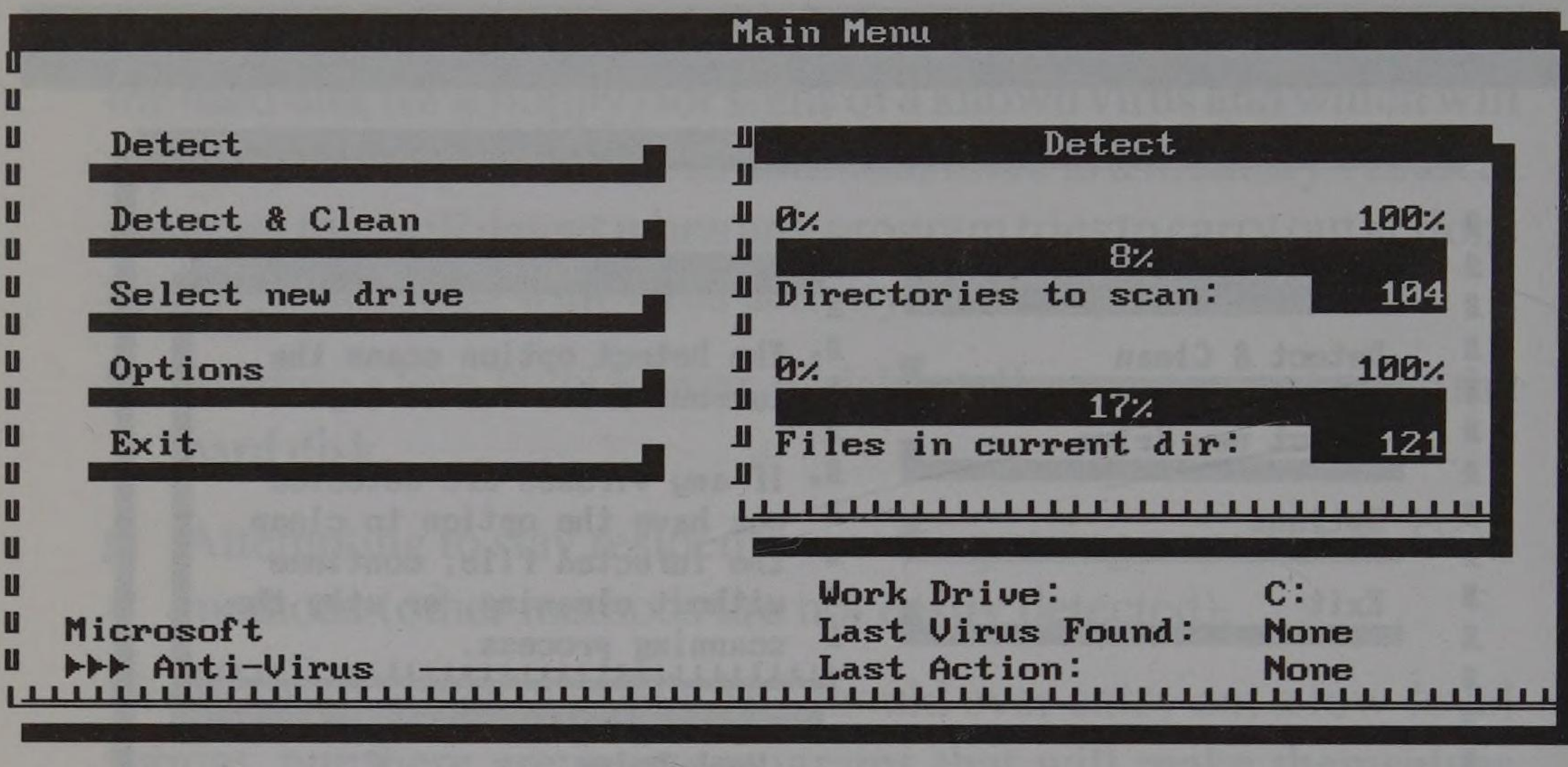
1. Start up the program called MSAV.EXE, which should be located in the C:\MSDOS directory. This brings up the **Main Menu**, and there will be a pause while the hard disk directories are checked.
2. Unless you know that there is a virus infection, click on the **Detect** option. You can use **Detect and Clean** if you are sure that an infection is present.
3. The memory will be scanned, followed by a scan of each file in each directory (screenshot on Page 254). This takes time.


Microsoft Anti-Virus 14:30
Main Menu
Detect
Detect & Clean
Select new drive
Options
Exit
Detect
• The Detect option scans the current drive for viruses.
• If any viruses are detected you have the option to clean the infected file, continue without cleaning, or stop the scanning process.
Microsoft
Anti-Virus
Work Drive: C:
Last Virus Found: None
Last Action: None
Help ♦Drive ↑Exit ↑Detect Clean ♂Delete ♀Options┐List

The main menu of MSAV, the DOS anti-virus scanner.

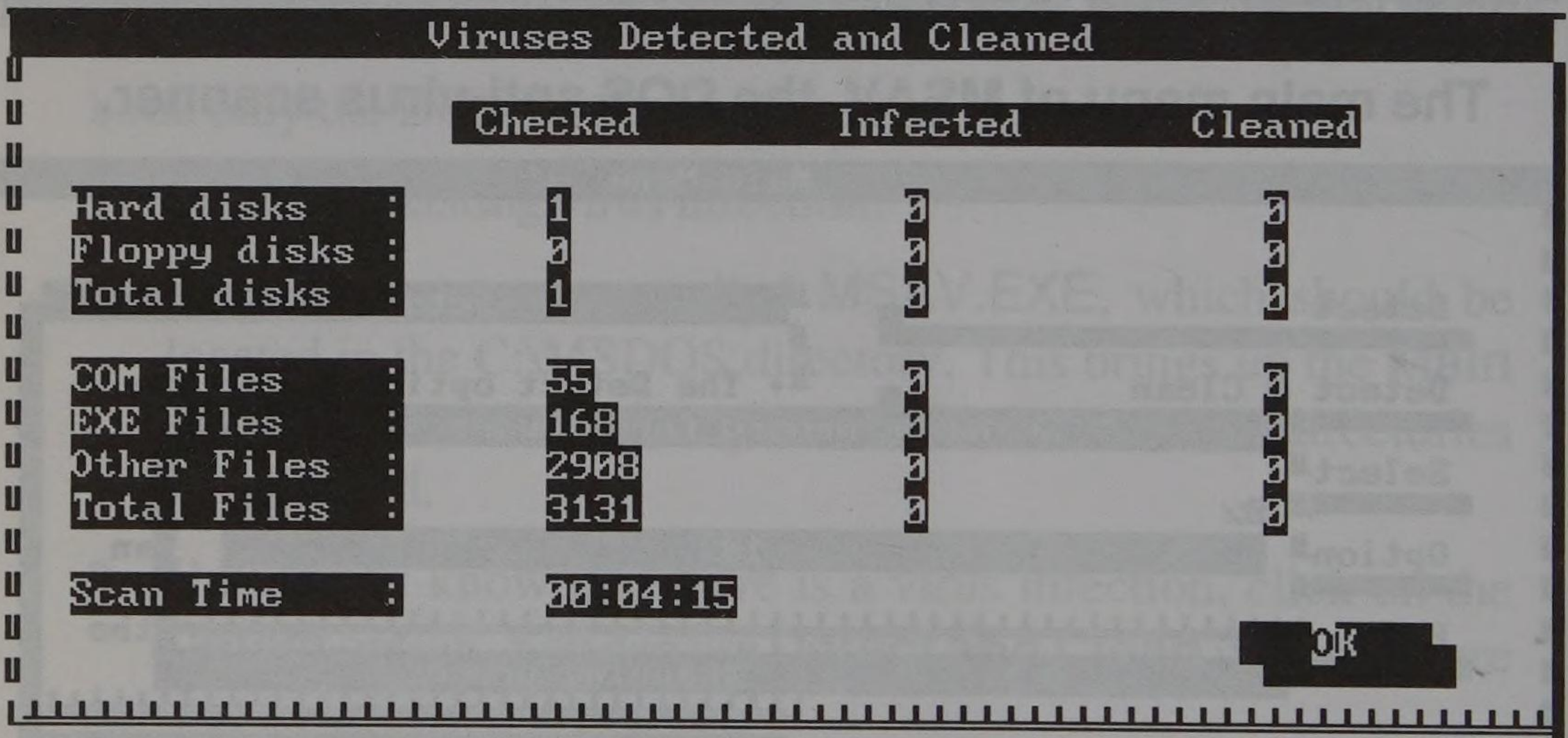


A memory scan in progress, the first MSAV action. The scale shows how far the scanning has reached.



Scanning the files and directories. This can take a fair time on a well-populated large disk.

4. When scanning is completed, a report will be issued.


Viruses Detected and Cleaned
Checked
Infected
Cleaned
Hard disks : 1 0 0
Floppy disks : 0 0 0
Total disks : 1 0 0
COM Files : 55 0 0
EXE Files : 168 0 0
Other Files : 2908 0 0
Total Files : 3131 0 0
Scan Time : 00:04:15
OK

A report from MSAV showing, in this example, no detected virus problems.

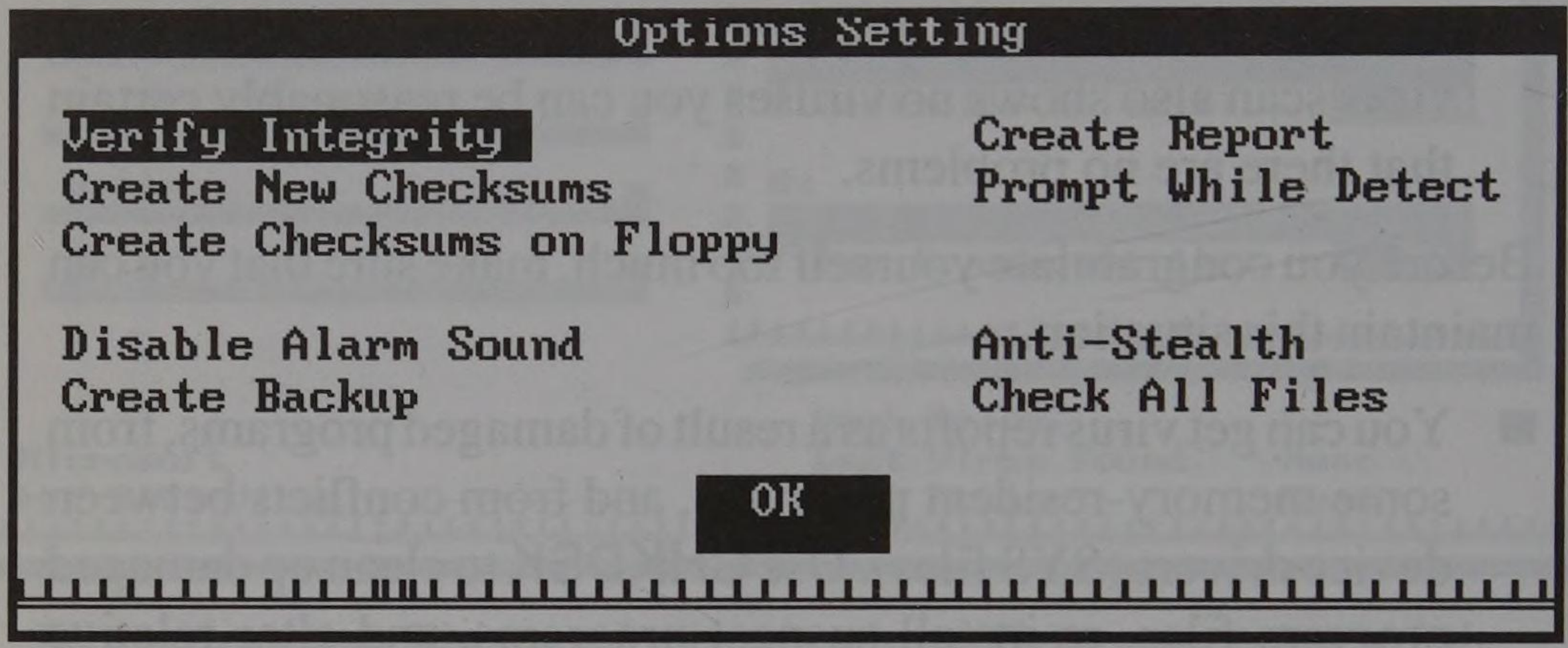
In this example, no viruses have been found, and because this machine is never connected to a telephone line nor used with games software or unsolicited disks the conclusion is hardly surprising.

- Remember, however, that a clean report from an anti-virus scan is not an absolute discharge - there are viruses that may be present yet not detected. If you see no traces of virus activity in your day-to-day use of the computer (see above) **and** the anti-virus scan also shows no viruses you can be reasonably certain that there are no problems.

Before you congratulate yourself too much, make sure that you can maintain this situation.

- You can get virus reports as a result of damaged programs, from some memory-resident programs, and from conflicts between device drivers (SYS files). Use **CHKDSK** to clean up damaged program files, re-install suspect programs, and alter relative position of driver lines in the CONFIG.SYS file before deciding to go on a full virus-buster rampage.

The Main Menu of Anti-Virus contains an **Options** section which, when clicked, produces the menu shown below.



The Options menu of Anti-Virus, showing the nine choices.

The sections of this menu are:

1. **Verify Integrity**. This is ticked as a default, and it looks for changes in files, using a Checksum (a number based on adding the number-codes of a file). The checksums for your files are created by the second option, also by default.

- This action can help to detect new viruses, contracted after this verification, which use methods as yet not known.

2. **Create New Checksums**. This option, switched on by default, creates a file called CHKLIST.MS that consists of a checksum for each file. There is a separate CHKLIST.MS for each directory on the hard disk, and when new files are placed in the directory each Anti-Virus scan will add data for the new files. These files are used in the comparisons made using Verify integrity.

3. **Create Checksums on Floppy** will carry out the creation

of a CHKLIST.MS file for a floppy disk if Create New Checksums is also selected for use.

- After this has been done for a floppy, the disk should be write-protected. The option is off by default.

4. **Disable Alarm Sound** is normally off. When ticked it allows the normal sounds to be disabled so that only the screen warnings remain.

5. **Create Backup** needs to be used with care - the default is to turn this option off. When on, its effect is to make a backup of any infected file before attempting to clean the file.

- This is a desperate measure for a program that is precious because you have no other backup, and the backup should be put on to a floppy. It will carry the **VIR** extension. Using this option means that when the virus has been removed from the original there will still be an infected copy.

6. **Create Report** is also switched off by default. When switched on it creates an ASCII text file placed in the root directory and containing a dated and timed report of all viruses found and removed.

7. **Prompt while Detect** is normally switched on, and produces three options in a dialogue box when a virus is found in a file. The options are to repair the file, continue scanning without repairing, or the stop scanning.

- With this option off, the scan detect and/or clean actions do not provide any messages until the end of the scan.

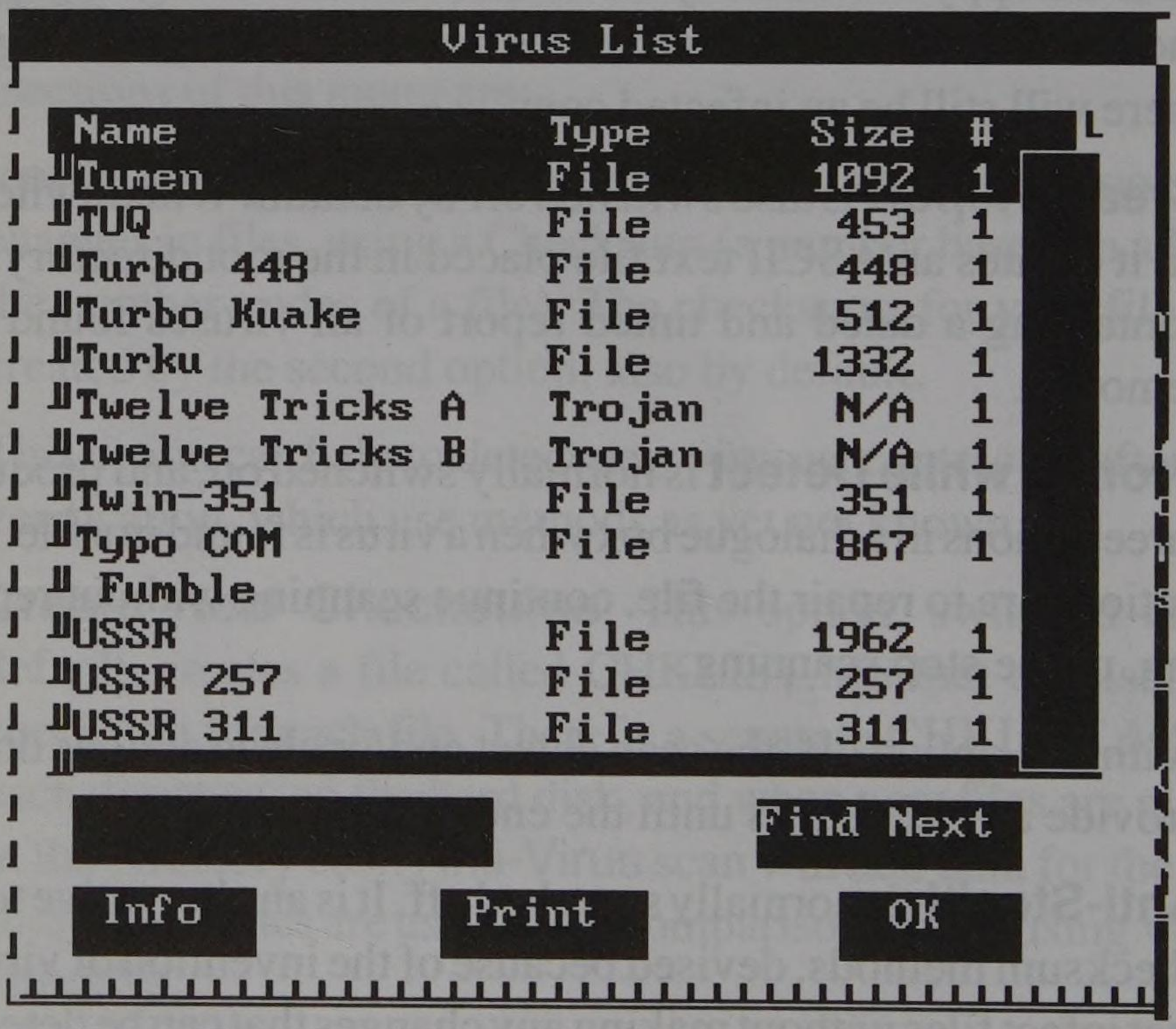
8. **Anti-Stealth** is normally switched off. It is an alternative to the checksum methods, devised because of the invention of viruses that infect files without making any changes that can be detected by simple checksum methods.

- Anti-Stealth is normally switched off because its action is slower, since it has to check files more thoroughly, but it can be used if you have used Verify Integrity without finding a virus but you have reason to suspect that a virus is still present.

9. **Check all Files** is normally on, so that both data and program files are checked. When this option is switched off, only files with the extensions EXE, COM, OVL, SYS, BIN, APP and CMD are checked. These, however, are the most likely to be infected.

- When the option is switched on, files with extensions 386, DOM, DLL, DRV, FON, ICO, OL*, PGM, PIF, PRG are also checked.

The settings that you make in the Options can be saved as a configuration file when you leave Anti-Virus. This ensures that the same settings will be used next time.

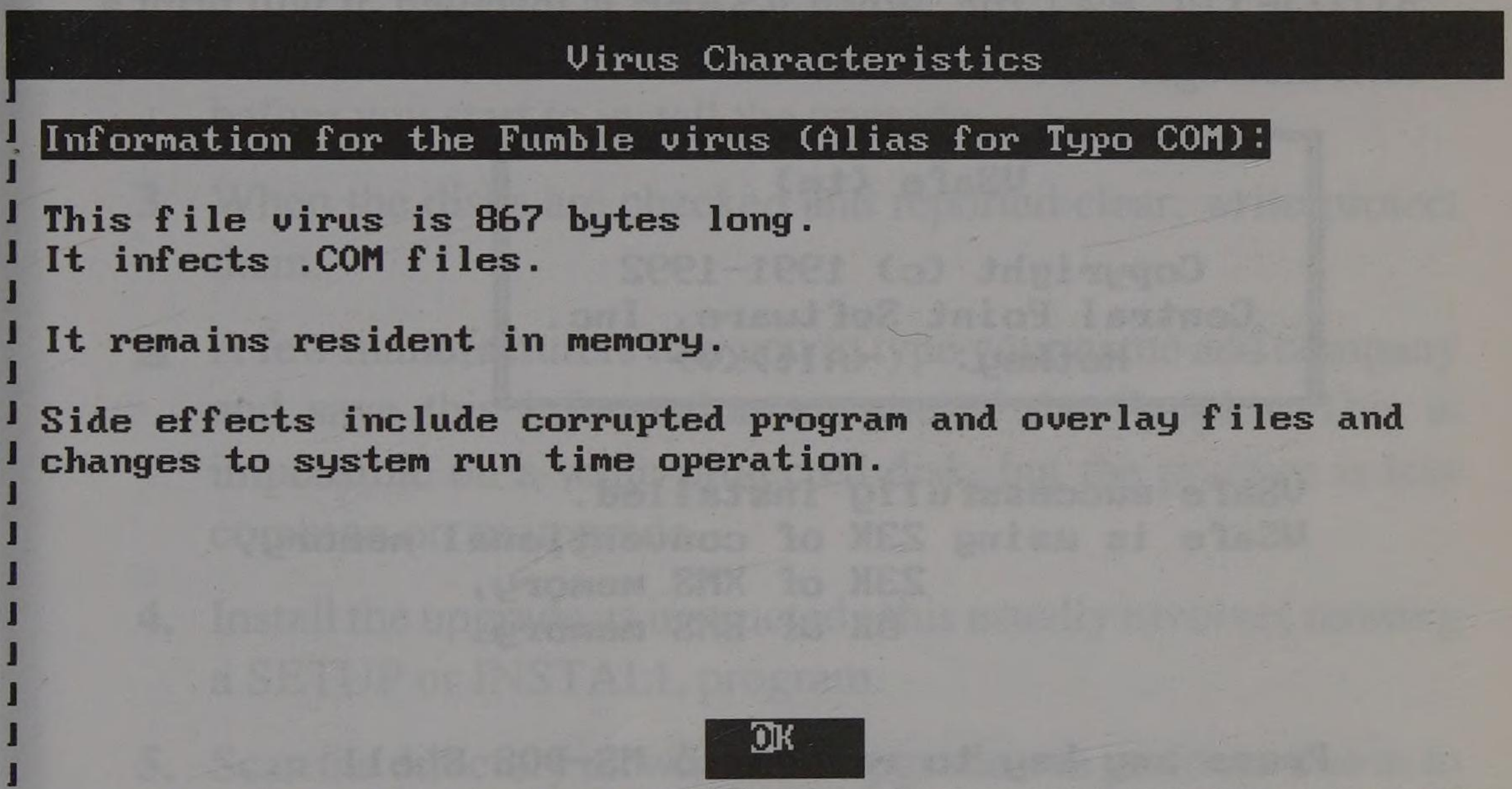


Part of the file of known viruses with their alias names and number of known aliases.

■ A set of command short-cuts is listed at the foot of the screen.

While Anti-Virus is running, you can look at a list of viruses. Press the **F9** key, or click on the **List** option at the bottom of the screen.

The illustration on the last page shows part of this (very large) list of known viruses. Clicking on a name will produce a detailed note, for that virus, showing size, what files are infected, whether or not there is any memory-residence, and the effects. The list window also provides for printing information.


Virus Characteristics
Information for the Fumble virus (Alias for Typo COM):
This file virus is 867 bytes long.
It infects .COM files.
It remains resident in memory.
Side effects include corrupted program and overlay files and changes to system run time operation.
OK

The further report on a particular virus

If your system is vulnerable to virus infection you can install a scan for viruses in the AUTOEXEC.BAT file. This will carry out a scan each time the computer is switched on, and requires you to add the line:

MSAV C: /n

- the **/n** option avoids the use of the display that you get when you run the **MSAV** program normally, and puts the results into a file.

The **MSAV** command word can be followed by a drive letter with its colon. There is a full set of MSAV command options detailed in the *Pocket Book of MS-DOS 6.0*, but inexperienced users are better served by keeping to the defaults.

VSAFE

Another line of protection is to try to detect when a virus is starting its activity. This can be done by using a memory-resident program called VSAFE, which would normally be added as a line in the AUTOEXEC.BAT file. When VSAFE is installed, it will print a brief message.

```
                    VSafe (tm)

              Copyright (c) 1991-1992
            Central Point Software, Inc.
                Hotkey:   <Alt><V>


VSafe successfully installed.
VSafe is using 23K of conventional memory,
               23K of XMS memory,
                0K of EMS memory.

      Press any key to return to MS-DOS Shell....
```

The VSafe installation notice.

- Note that some XMS (extended) memory is used, so that very little conventional memory is needed. The message shows that the **Alt-V** key combination will bring up a menu.
- This hotkey action cannot operate if you are using DOSSHELL - exit from DOSSHELL before using VSAFE options.

VSAFE can be used with a set of options, but, as usual, these are intended for the experienced user.

Living with Anti-Virus Protection 69

If you practise safe computing, as outlined at the start of this Part, you need not bother too much with Anti-Virus protection, and 99% of suspected virus problems are really due to the well-known problem of the erratic digit. If you are exposed to virus infection, however, and you make intensive use of the MS-DOS virus software, you will need to remember a few important points.

1. When you upgrade software you are likely to get virus messages because the upgrading will itself involve actions similar to this carried out by a virus. You should therefore proceed as follows:

2. Scan the floppy disks from which the upgrade will be made before you start to install the upgrade.

3. When the disks are checked and reported clear, write-protect them.

- A few manufacturers like you to type your name and company and save this information on one of the floppies. This is impossible on a write-protected disk, but the practice is less common on an upgrade.

4. Install the upgrade as instructed - this usually involves running a SETUP or INSTALL program.

5. Scan the directory on which the upgrade was performed so as to update the CHKLIST.MS file.

Sometimes a virus will have extensively damaged a file before it is detected. In such a case, Anti-Virus will deliver a message to the effect that the file is damaged beyond repair, and ask you if it can delete the file in order to prevent further infection.

- The normal answer will be **YES**, because you should have adequate backups.

You may also get an **Invalid Signature** message. This is usually an indication of a new virus about which Anti-Virus has no information.

69 Living with Anti-Virus Protection

The *signature* is a tell-tale set of numbers that are always present in a particular virus. You should register for regular updates to Anti-Virus.

VSAFE in particular will deliver messages that indicate attempted actions that can in some cases be caused by normal circumstances, and it is up to you to determine what are your normal circumstances.

1. If a network is loaded after starting VSAFE you are likely to find VSAFE warning you of attempts to modify system memory. If it is possible to load VSAFE after loading the network, do so. If you cannot do this, check with a virus scan the first time you are warned of memory modification.

   - If no virus is found it is likely that the normal action of the network software is the cause of the message. You can use the **Continue** option of VSAFE in future.

2. A warning of a program becoming memory resident should be treated as suspicious if you are not loading a memory-resident program.

   - Make sure that all the memory-resident programs you need are loaded before VSAFE, and make a note of any other memory-resident programs that you know you are likely to use.

3. Writing to a disk when you are not saving data is not always suspicious. When you use SMARTDRV, writing is usually carried out shortly after a save action, and some programs, notably word-processors, save their text file(s) automatically every ten minutes or so.

4. You will encounter problems if you have loaded other memory-resident programs following VSAFE, and you then remove VSAFE from memory. This will unload the programs that were loaded subsequently.

InterInk 70

InterInk is intended as a simple utility for connecting a laptop to a desktop computer, but it can be used as an elementary form of network system for two computers, using either parallel or serial ports. Most computers possess only one parallel port which is used for the printer, but it is quite common to fit two serial ports and even if one of these is used for a serial mouse this leaves one spare for InterInk connection.

> InterInk does not rank as a network, and if a program asks you during SETUP if you are running on a network, answer **NO** if you use only INTERLNK.

Since the use of serial links is much more common, this will be described. Serial links allow transfer rates of just over 115,000 bits per second, which is considerably slower than can be achieved with parallel ports, but fast enough for printing or file copying. The hardware consists simply of a seven-core serial non-modem cable terminated in the type of plugs that the machines use - either a 25 pin or a 9 pin type.

- Serial cables intended for connecting printers or external modems are useless - their connections are not suitable. You should specify that you want a non-modem (or null-modem) connected cable for linking two computers together. If a parallel cable is used it must be of the bidirectional machine-to-machine type - these are unusual.

With INTERLNK in use two machines can share disk drives and printers. One machine is designated as the **server** and the other as the **client**. The Client can be used normally, with the drives and printer of the server at its disposal. During this time the server is immobilised - you **cannot** use it normally.

InterInk can be used in a variety of configurations, and you need to decide for yourself, after trying some out, what will be best suited

for your own uses. InterInk can be used along with Windows and DOSSHELL on either machine, though task switching cannot be used while the machine is being used as a server.

The minimum requirement to use InterInk is to add INTERLNK.EXE in a CONFIG.SYS line to the machine you will use as a Client. You might want to use this line in the CONFIG.SYS files of **both** machines to allow yourself some flexibility. A typical CONFIG.SYS entry is:

DEVICE=C:\MSDOS\INTERLNK.EXE

which will install InterInk (whose files are, in this example, in the MSDOS directory) and look for a connection as the computer is starting up. This connection need not exist at that time and can be started at any time later by using **INTERLNK** or **INTERSVR**, see later.

InterInk Options

The options that can be used following INTERLNK.EXE are:

/DRIVES:n InterInk assumes three drives. If you have fewer or more than three on the **other** computer you can specify them as, say, **/DRIVES:2** or **/DRIVES:4**. If you use **/DRIVES:0** no drives will be shared, only printer connections.

/NOPRINTER Ensures that the printer is not shared. Remember that if a printer is shared both machines must be switched on and running InterInk before printing can be done from the machine that does not have the printer connected.

/COM Can be used in the form **/COM:1** to specify a serial port (usually in the range of **COM:1** to **COM:4**) rather than have Interlnk find it for itself.

/LPT Can be used in the form **/LPT:1** to specify a parallel port for data transfer (**NOT** for the printer).

/AUTO Creates a link only if the server is active, otherwise does not load InterInk into memory.

/NOSCAN Installs InterInk in memory, but does not attempt to link with the server.

/LOW Installs InterInk into conventional memory rather than the default upper memory.

/BAUD Used to specify a serial transfer rate if for some reason the normal rate of 115200 cannot be used. The slower rates are 57600, 38400, 19200 and 9600 and they will make access noticeably slower.

/V Prevents timer conflicts - use this if, for example, a serial mouse stops working when Interlnk is running.

When the INTERLNK.EXE line runs in the CONFIG.SYS file it establishes a set of new drive letters, equal to the number of drives on the other computer that can be shared. If the number of drives on the other machine is not the default 3, you should use the /DRIVES option to correct this. It is also useful to add the /NOSCAN option if you often start one machine alone or at a different time from the other.

- If you do not share a printer then add the **/NOPRINTER** option because this reduces the memory requirements of INTERLNK.
- The **/COM** option can also be useful, particularly if you use a serial mouse on the COM1 port. Using **/COM:2** in INTERLNK.EXE prevents the program from checking COM1 and possibly interfering with the mouse action.

INTERLNK and INTERSVR

The normal use of Interlnk involves starting the **INTERSVR** program on the Server machine and **INTERLNK** on the Client.

1. On the server machine, run **INTERSVR**, either direct from DOS or by way of DOSSHELL or Windows. This will show only the outline window when the other machine is not yet activated.

```
Microsoft InterInk Server Version 1.00
   This Computer          Other Computer
    (Server)                (Client)
   A:
   B:
   C: (229MB)
   F:
   G:
   H:
   I:
   J:
   K: (130Mb)
   D:
   E:
   LPT1
```

The INTERSVR screen as it appears before a connection is made.

2. On the Client machine, start **INTERLNK** either at the MS-DOS command line or by clicking on the command in DOSSHELL or Windows.

- If you are running DOSSHELL or Windows you can click on a drive letter that is linked to the other machine instead of running **INTERLNK**.

- If you have previously used **INTERLNK** under DOSSHELL there may be remote drive information held in memory, so that clicking on the remote drive letter produces an old directory. Use the **F5** key to refresh the display.

3. The screen of the client machine will show a list of equivalent drive letters and the display on the server machine will change to reflect the connection. The connection remains in use until it is broken by pressing **Alt-F4** at the server.

4. The server diagram remains on screen as a reminder of the connections. In the example, the C:\drive of the server machine is referred to as F: on the Client machine, so that a command such as:

   COPY F:\words*.TXT C:\text

   can be used. For most purposes it is easier to use DOSSHELL or Windows File Manager to carry out such actions.

   Running INTERLNK from DOSSHELL or Windows presents no problems, but when INTERSVR is run the INTERSVR menu remains on screen. You cannot make use of DOSSHELL actions like task switching while INTERSVR is running, and the same applies to Windows. A message will appear to remind you of this.

   You may have problems in some cases with different MS-DOS versions. Normally, provided that one machine is using MS-DOS 6.0, the other can be using any version from 3.0 onwards. This is **not** true if the server machine uses a single large disk partition, as is normal, and the Client is using MS-DOS 3.0 to 3.4, because these versions cannot make use of large disk partitions.

5. If you use the Client machine to run a program that is located on the disk of the Server you need to be sure that the program is one that could run on the Client (not configured for a different video system, for example).

6. You cannot use the following commands over an Interlink connection:

CHKDSK DEFRAG DISKCOMP FDISK
FORMAT SYS UNDELETE UNFORMAT

- Drives created by a Network are not handled by InterInk.

- 8 -

Command Summary

71 Other Commands

The main parts of this book have dealt with the important MS-DOS 6.0 commands and utilities in as much detail as the less-experienced user needs. This Part is devoted to a brief summary of other commands. Since these are of secondary importance, it is quite possible that you will never need to use some of them. The amount of space devoted to a command in this list does not reflect its importance. The most important commands are given very little space because they have already been dealt with in more detail.

- Some older commands are no longer supplied with MS-DOS 6.0, and have been omitted from the list. If your computer has previously been used with an older version of MS-DOS you might find these commands in the C:\or C:\MSDOS directory. You should delete old commands that are no longer contained in MS-DOS 6.0.

- Items such as option letters are not listed, because you can find what options are available, with example, by using the HELP command. For example, using **HELP COPY** will produce a set of data pages on the COPY command, including options and examples.

- Some utilities, like EDLIN are ridiculously out of date. Others, like DEBUG, are intended for programmers only. They have been omitted, along with batch file commands that are intended for programming purposes.

- The utilities that relate to the code page system for IBM units have been omitted, since so few users require them

- As always, if you need more information and are able to use it, you will find detailed instructions and notes in the *Pocket Book of MS-DOS 6.0* and in the MS-DOS 6.0 manuals.

APPEND

does for data files what PATH does for program files, allowing the machine to search through different drives or directories for the data file that you want. The command would be used in the AUTOEXEC.BAT file, but it is not much used nowadays.

Example: **APPEND \TEXT;\SPSHEET;\DBASE**

This will search these three directories for a data file.

ATTRIB

allows you to alter the readonly(**R**), archive(**A**), system(**S**) or hidden(**H**) setting of a files, or to report on the settings of these attributes.

Example: **ATTRIB BOOKTXT.TXT** produces on screen:

R A:\BOOKTXT.TXT

meaning that this file is readonly, it cannot be deleted or edited.

Example: **ATTRIB +R C:\LURK\KEEPIT.DOC**

will make the file KEEPIT.DOC in this subdirectory a readonly file.

Example: **ATTRIB R *.*/S**

will make any file on the disk which is readonly into readwrite, so that it can be deleted or edited.

CHDIR

or **CD** displays or changes the current directory, see Section 21.

CHKDSK

is a disk utility whose purpose is to check how a hard disk is used for storing files, and also to report and correct errors (in the form **CHKDSK /F**). Because of the way that CHKDSK uses memory, it

must not be used while the machine is running MicroSoft Windows or DOSSHELL.

If you find that using **DIR** produces nonsensical reports, like files of many Mbytes or names that consist of non-alphabetical characters, then you probably need to run **CHKDSK /F** .

CLS

clears the screen and restores normal video (bright on dark) display.

COPY

is a multipurpose command which can copy files to disk, to screen or printer, alter date and time of files, or can combine files into one single copy. See Sections 25 - 27 for details.

COUNTRY

is used in the CONFIG.SYS file, see Section 17.

DATE

allows the date to be set in the internal clock of the computer. **DATE** used alone will print the date and ask you to alter this if you want (press **ENTER** to leave it alone, or type a date and then press **ENTER**. Alternatively you can type the command DATE and follow it by a date. The date format is determined by the country file of MSDOS, and for the UK is DDMMYY.

DBLSPACE

starts the doublespace utility for managing compressed-data disks, see Section 62 for details.

DEFRAG

de-fragments files and can re-organise the allocation of space on the hard disk. See Section 61.

DEL

(or **ERASE**) can be used to delete files, either single files or, using a wildcard, in groups. See Section 29 for details.

DELTREE

will delete a directory along with all of its files and its sub-directories (and files in the sub-directories). The directory to be deleted must be specified, and cannot be the current directory. You will be prompted to confirm the deletion.

DEVICE

CONFIG.SYS command, see Section 17.

DIR

is the directory command, the method of finding what files are contained on a disk or a directory. See Section 20 for details.

DISKCOMP

is intended for floppy disk users only, and should not be applied to a hard disk, nor to networks. It allows two floppy disks to be compared, noting any differences. The command is by now obsolescent.

DISKCOPY

is used for floppy disks to make a complete copy of one disk to another disk of **identical** type. The destination disk should preferably be formatted and unused, but if it contains files, these will be overwritten. If it has not been formatted, then formatting will be carried out before copying but this takes longer.

DISKCOPY copies byte-by-byte with no regard for files, so that system tracks, hidden files, deleted files, corrupted files etc. will all

be copied across to the destination disk. The disks must be **fully compatible** in layout - you cannot use DISKCOPY when one disk is a 3.5" 720 Kbyte type and the other is a 5.25" 1.4 Mb type, for example. Always use DISKCOPY if you want to make an **exact** copy of a disk which contains files you want to work on with a disk utility program (if, for example, you want to undelete files, or try to fix a CRC error fault).

DOSHELP

displays a summary of Help for DOS commands. The command word should follow **DOSHELP**, with a space between. The other way of producing **DOSHELP** is to type the command word followed by /?. For example, for help on DIR you could use either **DOSHELP DIR** or **DIR /?**

EDIT

starts the editor program from the command line. EDIT is available only if the QBASIC.EXE program remains on the disk - if this program has been deleted to save space you cannot use EDIT. If you start EDIT without a filename you will see a reminder screen that you have to remove by pressing the **Esc** key and you then have to select **File - Open** to load a file.

Starting in the form:

EDIT C:\AUTOEXEC.BAT

will short-circuit all these preliminaries for editing a file that already exists or which you want to create.

EXIT

is used when a program has allowed a return to MS-DOS so as to allow you to make use of one or more MS-DOS commands. When the commands are finished, you should type **EXIT** (then press **ENTER** key) to return you to the program that you temporarily left.

EXPAND

is used to convert files in compressed form on the MS-DOS distribution disks into their normal usable format. No wildcards are permitted. Example:

EXPAND A:\DBLSPACE.EX_ C:\MSDOS\DBLSPACE.EXE

will expand and copy the file from the disk in the A drive to C:\MSDOS.

FASTOPEN

allows fast access to any file that has been used during a session, by temporarily storing the location that the file occupies on the disk into the memory of the computer. This can be used only with a hard disk system, and does not apply to networks or floppies FASTOPEN is intended to be used when the programs that you run are, in the main, compilers or databases. There is **no** particular advantage in using FASTOPEN along with other types of programs.

FC

is used to compare two files and display differences. **FC** can be used for ASCII (text data) files or binary files, but the main applications are to text files.

FDISK

is used to prepare an old-style MFM or RLL hard disk for formatting. Modern IDE drives are pre-formatted and you should not use FDISK on them.

FIND

is a text file utility or filter that allows each place where a set of characters (a string) has been used to be found in a file, a keyboard input, or when used as a filter the output of a program. These last

two options are valuable features which are seldom used. A large range of option letters can be used.

Example: **FIND "computer" C:\text\oldbook\chap1.txt**

will find each occurence of the word **computer** in the selected file.

FORMAT

prepares a floppy disk for use, or wipes a previously used disk clean. This is one of the few commands that (in its **/U** Unconditional option) can totally delete data beyond hope of recovery unless there is a Sentry file on the disk. It can be used for a hard disk, and though it will not necessarily destroy all of the data beyond recovery (depending on the disk partitioning), hard disk users should ensure that FORMAT is not present on the hard disk. Options other than **/U** and **/Q** (quickformat for disks that have been formatted before) are seldom used.

GRAPHICS

can be used for CGA, EGA or VGA screen systems to allow a screen-dump (printing on paper the screen appearance) for specified types of printers. It is seldom needed nowadays.

HELP

starts the MS-DOS Help system and, used alone, provides a list of Help topics. Clicking with the mouse on any topic will produce more detailed Help, with sections for Syntax, Notes and Examples. In most Help pages there are cross-referred topics that can be clicked on to produce more Help. The other method of using it is to follow the word with the command you want help on. For example:

HELP FC

will provide Help on the **FC** command, showing options and giving examples.

INTERLNK

will connect two computers together by way of a serial or parallel link, see Section 70. The command is executed on the client machine.

INTERSVR

is used to operate an INTERLNK connection on the server machine, see Section 70.

KEYB

is the keyboard driver program which can use a pair of letters to denote country

Example: **KEYB UK**

will select the UK version of the key layout, assuming that this is in the current directory.

KEYB is generally used in the AUTOEXEC.BAT file, Section 18.

LABEL

allows a name of up to 11 characters to be put into the disk directory as a volume name or label. The LABEL name will appear each time the directory for the disk is printed, as the VOLUME name. It is an advantage to use a label name for a hard disk, as this will cause a prompt for approval if an attempt is made to delete all files.

MEM

is used for memory diagnosis, see Section 53.

MEMMAKER

starts the memory diagnostic and optimisation program, Section 56

MKDIR

or **MD** Internal, creates a new directory, see Section 23

MODE

is a complex multipurpose command for altering settings relating to screen, printer, serial port and keyboard use. It is rarely required.

MORE

is used only as a filter in conjunction with the output of other programs.

MOVE

moves files from one drive or directory to another - this is a copy action followed by deletion of the original files. See Section 29.

MSAV

scans the computer disks and memory for known viruses. See Section 67.

MSBACKUP

controls the backup and restore actions, allowing a backup to be made of all or part of a hard disk. See Section 58 for details.

MSD

is a diagnostic utility that provides a detailed technical report on your computer. You might be asked to run MSD and send the disk file it created to a software supplier if a piece of software would not run correctly in your machine.

PATH

determines the directories that should be searched when a program name is typed. For more details, see Section 18

POWER

reduces power consumption on a portable or laptop machine when programs or devices are idle. This is useful only if the hardware conforms to the Advanced Power Management (APM) specification (as is normal nowadays). The CONFIG.SYS file must contain a line such as:

DEVICE=C:\MSDOS\POWER.EXE ADV:MAX

PRINT

is a command whose effect is to print an ASCII file of text on the printer. When you use **PRINT**, followed by the name of a file, the file will be printed on paper, but the computer can still be used for other tasks. PRINT is a timesharing action, one that is done by loading in text in batches from the disk, storing the text in memory, and sending the text to the printer in any spare time that the computer happens to have. PRINT is very seldom used, because word-processors and Editors use their own printer drivers.

PROMPT

is used to change the normal prompt from the form:

C:\> or **C:\MSDOS>**

to some other form, which can include date and time if needed.

QBASIC

starts the QBASIC interpreter running so that you can write or run programs in this language. For details, refer to specialised books, such as *The QBASIC Book (Ian Sinclair)* from Bruce Smith Books Ltd.

- Note that if you delete the QBASIC.EXE file from your disk you cannot use EDIT.

REM

is used in AUTOEXEC.BAT and CONFIG.SYS commands as a way of making a comment or preventing a command from being executed.

RENAME

(or **REN**) is used to change the name of a file, or a group of files if a suitable wildcard is used. A directory does **not** count as a file in this respect, and cannot be renamed by using the REN command. MSDOS can rename a directory by using the MOVE command, and DOSSHELL (see Sections 30 - 43) will also carry out this action.

REPLACE

is a seldom-used command that allows for another method of making selective backups and updates by replacing or adding files from one drive/directory to another. The command requires you to specify one path and filename, followed by a path that ends with a directory. The filename that is specified in the source drive or directory will then be used to replace a file of the same name in the second drive or directory, the destination. If no such file exists in the destination directory, the command stops with a message to this effect (**No files replaced**). A wildcard file specification can be used for the source file, so that several files can be replaced in a backup action.

RESTORE

replaces files to the hard disk that were backed up to floppy disks by using the old BACKUP command, see Section 58. The modern MSBACKUP is a replacement.

RMDIR

(or **RD**) removes a directory from a hard disk, see Section 24.

SET

is used to set environment variables. These are assignments that can be made in advance of running a program that uses them. The use of SET depends on the programs that you run, and may not be necessary for many of your programs. The form of the command is **SET *name=text***, where the ***name*** is one that will be used to pass the ***text*** to a program. There **must not** be any space on either side of the = sign.

For example: **SET TMP=D:**

means that a program which uses **TMP** to represent temporary storage space can make use of a RAMdisk or hard drive that has been set up as drive **D:**

SETVER

allows MSDOS 5.0 or 6.0 to pass an earlier version number to another program. Several programs read the version number of MSDOS and will not run correctly if the wrong version number is read - many such programs did not allow for version numbers beyond 3 in the belief that this was the ultimate in MS-DOS.

SHARE

can be used for networked systems so that files can be shared by each computer on the network. In theory it should be used along with Windows, but this is normally unnecessary.

SMARTDRV

loads and runs the Smartdrive disk cache program, Sections 63,64.

SORT

is used mainly as a filter, requiring input and output to be specified. See Section 47.

SYS

transfers the hidden MSDOS.SYS and IO.SYS files and COMMAND.COM to a formatted disk provided that room has been made for them during formatting. This is hardly ever required nowadays.

TIME

is used to set the computer clock, or to report the current time.

Example: **TIME**

gives the current time, and asks you to enter a new time. Press **ENTER** if you do not want to change the time.

Example: **TIME 15:15**

changes the current time to 3.15 p.m. The setting is made when the **ENTER** key is pressed on the typed time number, so that it is possible to synchronise the setting to the seconds hand of a watch or to telephone or TV time signals.

Note that the timekeeping of a PC is very poor because the count of the (quartz) crystal is done using software that can be interrupted by such actions as mouse movements.

TREE

displays directories and subdirectories. To print the output your printer should be able to cope with the IBM graphics character set most dot-matrix printers can be arranged to emulate the IBM ProPrinter, and HP Laserjet or compatibles can be set for the PC8 characters.

TREE C: /F will cause the names of files in each directory of the C: drive to be displayed.

TYPE

is the command that allows a text file to be copied to the screen. The format is **TYPE** followed by a space and the filename (and path). Wildcards are completely ignored.

Example: **TYPE C:\WORDS\FILE1.DOC**

UNDELETE

is used for restoring deleted files. See Section 60 for details.

UNFORMAT

is used to restore a hard disk to the state it had before a **FORMAT** (but not a **FORMAT/U**) command was executed. See Section 59 for details.

VER

is used to print on the screen the version of MSDOS that is currently being used.

VERIFY

has to be followed by **ON** or **OFF**. When **VERIFY ON** has been used, any file that is saved on a disk will be checked to ensure that the copy has been made correctly. The checking is **not** character-by-

character, but the sum of ASCII codes of characters (a checksum) so that it is possible that the copy might be incorrect due to two errors cancelling each other out, but this is most unlikely.

VOL

is used to print on screen the label of a disk (see LABEL), which is the name of up to 11 characters that you give to the disk. This applies mainly to floppy disks, and allows you to recognise your disks even if any paper labels have dropped off from the outside.

VSAFE

monitors the computer for virus infection and displays a warning if it detects the type of action that is used by a virus. See Sections 67 - 69 for details.

XCOPY

is an enhanced file copier which allows for selective copying of files that have archive bits set, or according to date. The main advantage of XCOPY, however, is that it can create a copy which has the same directory structure as the original disk. As XCOPY tests the archive attribute of a file (see ATTRIB), it can be used in a batch file to ensure that files that have just been created are correctly backed up without backing up files that have **not** been altered.

The **XCOPY** command is followed by the source and the destination drives and paths (if applicable), along with up to 8 option letters if these are needed. Wildcards are permitted for the source files.

Example: **XCOPY C:\WORDS*.* A:**

Symbols

@ECHO OFF 187
640 Kbyte limit 190
80x86 processor types 40

A

Actions of MS-DOS 14
Active Task List 143
Add Group window 135
Additional memory 190
Advanced Properties 129
Advanced Setup 142
All Files list 103
Alt key 143
Alt-Esc 146
Alt-Tab 146
Anti-Virus options 256
Anti-Virus system 252
APPEND 271
Appending to files 165
Application Shortcut Key 129, 138
Applications 19
Archive attribute 22
ASCII code 79
Associating data files 120
ATTRIB 271
Attributes 22
AUTOEXEC.BAT 53

B

Backslash 37
Backup 42
Backup modes 213
Backup Set Catalogue 216
Bad or missing keyboard file 55
Batch file 172
Batch file, creating 173
Batch file directory 179
Batch file, stopping 178
Batch files, uses of 181
Begin Optimization 230
Blocks of filenames 99
Bombs 246
Boot Sector virus 246
Booting 41
Bus mouse 44
Byte 15

C

Cache 238
Cache sizes, Initial & Windows 242
CALL 187
CD 65, 271
Changing drives 98
CHDIR 65, 271
Child directory 71
CHKDSK 271
CHKDSK /F 228
Clicking, mouse 92

Index

Clicking on a directory 95
Client 263
Closing down a directory 74
CLS 272
Collapsing the tree 102
Color options 154
COM files 37
Combining files 80
Command, internal 20
Command line 33
Commands line 128, 138
Commands, recalled 158
Commands, repeating 158
Compatibility test 208
Compressed files 44
CON 79
CONFIG.SYS 50, 194, 200
Confirm on Delete 150
Confirm on Mouse Operation 151
Confirm on Replace 150
Confirmation options 150
Conflict of names 183
Conflicts of memory use 196
Conservative settings, MEMMAKER 205
Contiguous 227
CONTROL.FIL 226
COPY 76, 272
Copy, with mouse 116
Copying 19
COUNTRY 30, 272
CP/M 12
Creating a batch file 173

Ctrl-C keys 178
Cursor, DOSSHELL screen 94

D

Data files 19
DATE 272
DBLSPACE 46, 233, 272
DEBUG 270
Default format, DIR lists 163
Default Parameters 133
Defaults, SMARTDRV 241
DEFRAG 227, 272
DEFRAG options 231
DEL 87, 273
Delete, with mouse 116
Deleting files 19, 87
Deletion, unintended 220
DELTREE 74, 273
Deltree 88
Deselecting 109
Deselecting blocks 110
Destination, in COPY 76
DEVICE command 273
Devices 79
DEVICEHIGH 201
Diagnostic reports 45
Differential backup 213
DIR 59, 273
DIR, default format 163
DIRCMD 161
Directory 21

Index

Directory, closing down 74
Directory Tree window 93
Disk actions 14
Disk cache 238
Disk space, installation 46
Disk Utilities Group 126
DISKCOMP 273
DISKCOPY 273
Display option, MSBackup 214
Display options, DOSSHELL 153
DOS 12
DOS Undelete 220
DOSHELP 274
DOSSHELL 92
DOSSHELL.INI 94
Dot symbol 36, 70
Double-clicking, mouse 92
Doubledot, in directories 70
Dragging files 118
Dragging, mouse 92
Drive letter 35
Drive letter selection 98
Drives, changing 98
Dual file list 102

E

ECHO 187
EDIT 274
Editing 26
EDLIN 270
EMM386 199

Enable Task Swapper 144
ENTER key 33
ERASE 273
Esc key, in File View 114
EXE files 37
Existing Groups 126
EXIT 274
EXPAND 196, 275
Expanded memory 191
Expanding the tree 101
Extended memory 191
Extension 35, 36

F

F9 key 113
FASTOPEN 275
FC 275
FDISK 275
File actions 14
File display 105
File Display Options 105, 152
File information 24
File Search 115
Filenames 21, 33
Filenames, drive letter 35
Filenames, extension 35
Files, data 19
Files, program 19
Files, selected 107
Files window 97
Filters 30, 168

Index

FIND 168, 275
FIND Options 169
Floppy disk 15
Floptical drives 17
FORMAT 276
Formatting 27
Fragmentation 227
Fragmentation report 229
Full directory path 35
Function key actions 159

G

Genius mouse 44
GRAPHICS 276
Groups 125
Groups, existing 126
Groups, running a program 142
Guidelines, viruses 248

H

Hard disk drives 16
HELP 276
Help Text line 130
Hexadecimal numbers, in File View 113
Hidden attribute 22
Higher-resolution displays 153
HIMEM.SYS 191

Index

I

IBM PC 13
Icon 92
IDE 206
Incremental backup 213
Initial cache size 242
Input redirection 166
Insert mode, DOSKEY 159
Interactive use 172
INTERLNK 263, 277
InterInk Options 264
Internal command 20
INTERSVR 265, 277
Invalid Signature 261

K

Kbyte 16
KEYB 55, 277
Kildall, Gary 13
Kilobyte 16

L

LABEL 277
Label name 59
Loader program 12
LOADHIGH 201
Loading 15
Locking up, and swapping 148
Logged 70

Logic bomb 247
Low-level format 252
Lower-case 34
LPT1 79

M

Machine configuration 207
Macro commands 160
Magnetic recording 15
Mbyte 16
MD 65, 278
Megabyte 16
MEM 277
MEMMAKER 202, 277
Memory, additional 190
Memory managers 194
Memory resident viruses 252
Memory use, conflicts 196
Memory-resident programs 192
MFM 206
MicroSoft 13
MKDIR 65, 278
MODE 278
Modern chips 40
MORE 170, 278
MOUNT 233
Mouse 44
Mouse actions 92
Mouse, types 44
Mouse methods 116
Mouse, serial 44

MOVE 88, 278
Move, with mouse 117
Moving 19
MS-DOS actions 14
MS-DOS directory 44
MSAV 252, 278
MSBACKUP 278
MSBackup utility 206
MSD 278
Multiple commands 139

N

Name, as parameter 184
Names, for files 35
Network 43
Network and VSAFE 262
New Program Object 134
NOPRINTER, INTERLNK option 265
NUL 79
NUL device 165

O

Open, DOSSHELL 124
Operating system 12
Options, Anti-Virus 256
Options, InterInk 264
Options menu, DOSSHELL 150
Options, SMARTDRV 242
OS 12
Output redirection 164

Overlays 16
Overtype mode, DOSKEY 159

P

Parameters 37, 132, 184
Parameters, absence of 186
Parasitic virus 246
Parent directory 71
Partial Backup 213
Password 129
password protection, MSBackup 214
PATH 54, 278
PATH line, AUTOEXEC.BAT 180
Pause after Exit 129, 139
PCTRACKR.DEL 226
Pentium chip 40
Permanent swap file 235
Pipe 30
Placeholder 184
portable computers 13
POWER 279
Preview option, display 153
PRINT 279
Printer 26
Printer drivers 26
PRN 79
Processor Speed test 208
Program files 19
Program Group 134
Program groups 97
Program Information 132

Program Item 134
Program Item Properties (PIP) 127, 134
Program options 46
Program options, installation 46
Program Title 128
PROMPT command 279
Prompt 30, 32
Prompt Message 133
Properties, Advanced 129

Q

QBASIC 279
Quit option, MSBACKUP 213

R

Range of selection 110
RD 65, 74, 280
Read-only 22
Read/write cache 239
README.TXT 42
Readonly cache 239
Reboot 41
Recalled commands 158
Redirection 164
Reformatting 217
REM 280
REN 84, 280
RENAME 84, 280
Renaming 21
Repeating commands 158

REPLACE 280
Reserved combinations, shortcut keys 139
Restore 216
RESTORE command 206, 280
RETURN key 33
RLL 206
RMDIR 65, 75, 280
ROM 12
Root directory 60
RPED editor 173
Run, DOSSHELL 124
Running and swapping 145
Running programs 32, 124
Running programs, from batch files 181

S

Saving 15
Scrolling windows 97
Search 115
Sectors 27
Select Across Directories 111, 152
Selected file 107
Sentry 220
Serial links 263
Serial mouse 44
Serial number 59
Server 263
SET 281
SETUP program 17, 45
SETVER 281
SHARE 281

SHIFT command 186
Shortcut keys, reserved combinations 139
Show Information, DOSSHELL option 152
Signature, of viruses 262
SIMM 191
Single file list 99
Slashmark 37
SMARTDRV 239, 281
SMARTDRV defaults 241
SMARTDRV options 242
SMARTDRV problems 240
SORT 171, 281
Sort order, directories 24, 108
Source, in COPY 76
Space, as separator 186
Special option, MSBackup 214
Startup Directory 128
Startup Directory line 138
Status box, directories 100
Sub-directories 65
Swap file, permanent 235
SYS 282
System attribute 22
System disks 27, 28, 48

T

Task Swapper, enabling 144
Task swapping 143
Temporary files 167
TIME 282
Time-bomb 247

Index

Total backup 212
Tracker 220
Tracks 27
Transfer rates 263
TREE 67, 283
Tree diagram 42
Trojan Horse 246
TSR 50
TYPE 44, 283

U

UMB 199
Uncompressed space 235
UNDELETE 220, 283
UNDELETE options 224
Underscore, in filenames 44
UNFORMAT 283
UNFORMAT utility 217
Unformatted disks 27
Uninstall 42
UNINSTALL disk 45
UNMOUNT 233
Upper-case 34
Utilities 14

V

Variables 161
VER 283
VERIFY 283
View file contents 113

Virus diagnosis **249**
Viruses **28, 246**
Viruses, guidelines **248**
Viruses, known **259**
VOL **284**
VSAFE **260, 284**
VSAFE and networks **262**

W

Wildcard **62**
Window Title **132**
Windows **14**
Windows cache size **242**
Worm **246**
Write-protection **28**

X

XCOPY **284**

Index

Virus diagnosis 249
Viruses 28, 246
Viruses, guidelines 248
VOI 284
VSAFE 250, 284
VSAFE and networks 252

W

Windows 62
Windows, Vista 16
Windows operating sites 242
Word 246
Write-protection 250

X

XCOPY 284

PC PLUS

STEP by STEP

MS-DOS 6.0

Do you need quick, straightforward and concise information about **MS-DOS 6.0**? Do you need it presented in an understandable step-by-step way? Do you want to see lots of examples, and find everything in easy to read sections? ...
...Then **this book is for you!**

This book

- Explains the purpose of **MS-DOS** and why it is still an essential feature of machines in these days of Windows - **step by step**
- Shows how to install **MS-DOS 6.0**, and explains how to arrange the installation for most convenient use - **step by step**
- Shows that only a few commands are essential to know and demonstrates their use, pointing you in the right direction for mastering other less essential commands - **step by step**
- Shows how to set up the memory of the 386 or 486 machine to best advantage, and compress disk information to make more of your hard disk - **step by step**
- Shows how to use DOSSHELL and get task-switching without Windows - **step by step**

If you are a beginner with a 386 or 486 machine, or if your last encounter with MS-DOS was using an old version - this book will show you how to cope with **MS-DOS 6.0** - **step by step**

KN-991-982

£12.95

NEW·TECH
An imprint of Butterworth-Heinemann Ltd
Published in association with
Britain's best-selling PC magazine PC PLUS

ISBN 0-7506-1726-8